VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

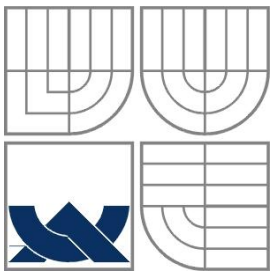
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

VYHLEDÁNÍ PODOBNÝCH OBRÁZKŮ POMOCÍ POPISU BAREVNÝM HISTOGRAMEM

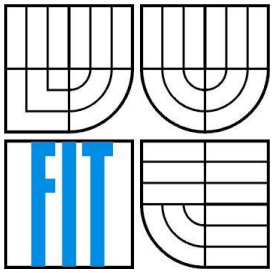
DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE BC. ZBYNĚK SAILER
AUTHOR

BRNO 2012



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

VYHLEDÁNÍ PODOBNÝCH OBRÁZKŮ POMOCÍ POPISU BAREVNÝM HISTOGRAMEM

IMAGE RETRIEVAL BASED ON COLOR HISTOGRAMS

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — BC. ZBYNĚK SAILER
AUTHOR

VEDOUCÍ PRÁCE — ING. VÍTĚZSLAV BERAN, PH.D.
SUPERVISOR

BRNO 2012

Abstrakt

Práce je postavena na popisu existujících metod vyhledávání podobných obrázků. Obsahuje souhrn způsobů popisu obrazu a kódování globálního i lokálního popisu (SIFT, atd.). Dále se věnuje způsobu efektivního vyhledávání v mnohorozměrném prostoru (LSH). Vlastní práce pak pokračuje návrhem a otestováním tří globálních deskriptorů využívajících barevné histogramy, histogram gradientů a kombinaci obou variant. Poslední část se věnuje vyhledávání podobných obrázků s využitím navržených deskriptorů a indexační metody LSH a porovnáním výsledků s existující metodou. Výsledkem práce je experimentální aplikace demonstrující navržené řešení.

Abstract

This thesis deals with description of existing methods of image retrieval. It contains set of methods for image description, coding of global and local descriptor (SIFT, etc.) and describes method of effective searching in multidimensional space (LSH). It continues with proposal and testing of three global descriptors using color histograms, histogram of gradients and the combination of both. The last part deals with similar image retrieval using proposed descriptors and the indexing method LSH and compares the results with the existing method. Product of this work is an experimental application which demonstrates the proposed solution.

Klíčová slova

RGB, oponentní barevný model, gradient, histogram, barevný histogram, deskriptor, vyhledávání, index, LSH, obraz, video, podobnost

Keywords

RGB, opponent color model, gradient, histogram, color histogram, descriptor, image retrieval, index, LSH, image, video, similarity

Citace

Zbyněk Sailer: Vyhledání podobných obrázků pomocí popisu barevným histogramem, Brno, FIT VUT v Brně, 2012

Vyhledání podobných obrázků pomocí popisu barevným histogramem

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením Ing. Vítězslava Berana, Ph.D. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

……

Bc. Zbyněk Sailer

21. května 2012

Poděkování

Děkuji vedoucímu práce Ing. Vítězslavu Beranovi, Ph.D., za vstřícnost, připomínky při konzultacích, podněty a odborné vedení diplomové práce.

Obsah

1 Úvod

Informace byly vždy velice důležité, stejně jako práce s nimi. S nástupem informačních technologií se jejich zpracovávání stalo v mnohém jednodušší než dříve a díky internetu se zjednodušilo i jejich šíření. Aby ale bylo možné získat potřebnou informaci, je třeba umět vyhledávat. Dříve bylo hledání například otázkou získání správné knihy a podle obsahu, či rejstříku vyhledání určité kapitoly, nebo strany. To ve světě informačních technologií není dost dobře možné. Webové stránky, a dokumenty jsou rozmístěné po celém světě, je jich téměř nepředstavitelné množství a mnohdy ani nevíme, kde bychom měli začít hledat.

Toto vyřešily takzvané vyhledávače. Ty umožňují vyhledávat jakékoliv dokumenty přístupné na internetu na základě určitých klíčových slov, či vět. Díky nim dnes v podstatě není problém získat jakoukoliv informaci.

Vyhledávání samozřejmě není doménou pouze internetu, ale v podstatě jakékoliv „databáze informací“, ať už se jedná o elektronickou knihovnu, sbírku hudby, nebo elektronická zařízení jako osobní počítač, či mobilní telefon. Informace jsou všude kolem nás a je jich čím dál tím větší množství. Z toho důvodu nám nezbývá nic jiného než naučit se informace vyhledávat.

Z důvodu velkého množství a různorodosti informací plyne, že vyhledávání musí být rychlé a hlavně efektivní. Znamení to, že vyhledaná informace by měla být co nejvíce relevantní. Z toho důvodu se vyhledávání neustále vylepšuje. Textové vyhledávání se například snaží využít kontextu a celkového „pochopení“ dotazu.

Informace se dnes vyhledávají téměř výhradně na základě textových „dotazů“. Toto řešení je efektivní, relativně snadno pochopitelné a pokrývající většinu potřeb. Zejména proto, že velké množství informací je textového charakteru.

Na druhou stranu je mnoho informací také grafického charakteru ve formě obrázků, či videí. Samozřejmě dnes není problém vyhledat ani grafickou informaci, jejich vyhledávání ale stále probíhá na základě textových „dotazů“. Ke grafickým datům je nezbytné dodat i textové informace (název, popis, apod.) aby je bylo možné vyhledat na základě textového „dotazu“. Pokud tyto informace chybí (databáze obrázků s anonymními názvy), není obvykle vyhledávání možné.

Proto se hledají další způsoby, jak vyhledávat obrazovou informaci. Jedním z těchto způsobů je vyhledávání na základě jiné obrazové informace. Příkladem může být vyhledání fotografií, na kterých se vyskytuje určitý objekt, či osoba.

Jednou z aplikací takového vyhledávání je hledání podobných obrázků. Způsobů, jak k této problematice přistupovat je mnoho. V podstatě se ale jedná stále o stejný problém. Je třeba obraz popsat, a to co nejjednodušším a zároveň co nejrobustnějším způsobem, aby bylo vyhledávání rychlé a co nejkvalitnější. Neboli aby výsledky vyhledávání byly co nejvíce relevantní. Takovým popisem obrazu a vyhledáním podobných obrázků na základě tohoto popisu se zabývá tato práce.

Cílem práce je navrhnout metodu popisu obrazu s využitím barevných histogramů, otestovat tuto metodu a využít ji pro vyhledávání podobných obrázků s využitím vhodné indexační metody. V semestrálním projektu byla na základě teoretických poznatků navržena metoda, která byla dále rozvíjena a upravována na základě poznatků z jejího testování.

V první části práce je popsáno, jak je obraz v počítači reprezentován, a možné způsoby jeho zpracování a popisu. Za tímto účelem popisuji barevné modely vhodné na ukládání a zpracování obrazu. Dále popisuji možnosti zpracování a popisu obrazu jako jsou histogramy, gradienty a jejich využití pro takzvané deskriptory. Popis kódování informací o obraze je rozdělen podle přístupu na globální a lokální.

V další části se věnuji popisu metod porovnávání deskriptorů a způsobů efektivního ukládání (indexace) deskriptorů, které umožňují vyhledávání prostorově blízkých bodů (nearest neighbour) v mnohorozměrných prostorech. Vyšší pozornost věnuji metodě LSH (Locality Sensitive Hashing), která poskytuje efektivní indexaci v prostorech s vysokou dimenzí.

V kapitole 4, Návrh řešení, popisuji návrh vlastních globálních deskriptorů využívajících barevné histogramy, histogram gradientů a jejich kombinaci (gradient + barevné histogramy). Vzhledem k zaměření práce se ale nejvíce věnuji deskriptoru využívajícímu barevné histogramy. Popisuji zde také existující metodu [6], ze které můj návrh vychází a kterou používám při testování jako referenční metodu.

Kapitola 6 popisuje datové sady a experimenty testující vlastnosti navržených deskriptorů v porovnání s referenčním deskriptorem (deskriptor z článku [6]). Následuje testování vyhledávání s využitím těchto deskriptorů s využitím další sady dat, která je zde rovněž popsána. Jako soubory obrázků jsem zde použil vhodně modifikovaná videa, jelikož globální deskriptory jsou vhodné pro vyhledávání velmi podobných obrázků a tuto posobnost právě splňují například sousední snímky ve videu.

Poslední kapitola se zabývá popisem vytvořené aplikace, která demonstruje použití metod vyhledávání podobných obrázků a to i pro porovnání dvou videí a vyhledání podobných, či stejných segmentů. Tato aplikace zároveň obsahuje algoritmy využité pro testování metody.

V závěru zhodnocuji dosažené výsledky a navrhuji možná vylepšení deskriptorů a další možné využití aplikace.

2 Popis obrazu

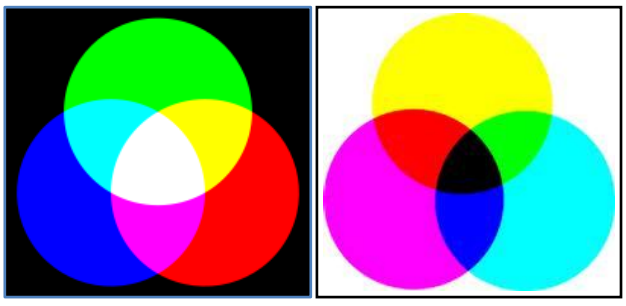
Tato kapitola se zabývá různými způsoby popisu obrazu, které se dají využít při hledání podobných obrázků. Popisuje některé barevné modely a rozděluje popis obrazu na globální a lokální.

2.1 Barevné modely

Fyzikálně je barva definována vlnovou délkou světla. Barvy které jsme schopni vnímat (takzvané viditelné spektrum) jsou tvořeny směsí světla různých vlnových délek v rozmezí cca 380 – 800 nm. Pro digitální zpracování je ale třeba barvu nějak reprezentovat (kódovat). Tato reprezentace musí být co nejjednodušší (typicky 3 složky) a zároveň co nejrobustnější, neboli co nejlépe popisovat barevné spektrum. Taková reprezentace je ale vždy určitým zjednodušením reality, nemůže realitu popisovat naprosto přesně. Z toho důvodu vzniklo mnoho způsobů digitální reprezentace barev zaměřených na určitou oblast využití. Některé jsou zaměřeny na elektronické zobrazování (RGB), jiné na fyzickou reprodukci tiskem (CMYK) a některé se s výhodou využívají pro zpracování obrazu (HSV, oponentní modely, ...).

Barevné modely se dají podle způsobu skládání barev rozdělit do dvou skupin na aditivní a subtraktivní.

Aditivní modely pracují na principu sčítání barev (vlnových délek). Barevné složky se sčítají a výsledkem je světlo větší intenzity než jednotlivé složky. Základním zástupcem aditivního barevného modelu (a zároveň nejpoužívanějším modelem) je RGB. Aditivní skládání barev se používá například při zobrazování na barevných monitorech.



Obrázek 2.1 – Aditivní (vlevo) a subtraktivní (vpravo) skládání barev

Subtraktivní modely pracují naopak na principu odečítání barev. Každá barevná složka „ubere“ určité vlnové délky světla a zbylé vlnové délky tedy určují výslednou barvu. Tohoto principu se používá hlavně při tisku barevnými pigmenty. Nejznámější zástupce subtraktivního skládání barev je model CMY (CMYK). Tyto modely ale nemají v mé další práci využití.

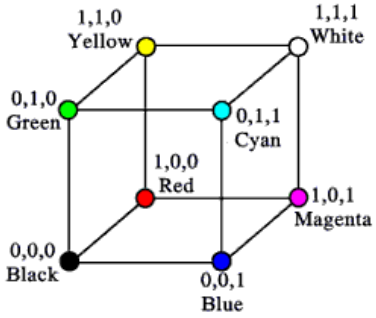
Barevný model RGB

Tento barevný model je nejznámější a nejpoužívanější. Využívá principu aditivního skládání barev. Vznikl na základě poznatku, že lidské oko je citlivé na tři základní barvy. A sice červenou (Red), zelenou (Green) a modrou (Blue). Odtud označení RGB.

Tohoto principu je využíváno i v digitální fotografii. Světlo je pomocí barevných filtrů rozloženo na tři základní barvy, které se samostatně zaznamenají (analogie k lidskému oku, které má tří druhy receptorů citlivých na různé vlnové délky světla). Naopak monitory mají jednotlivé body rozděleny na menší části (takzvané subpixely). Každý ze subpixelů zobrazuje jednu z barev RGB a složením intenzit jejich barev vzniká výsledný vizuální vjem (složená barva).

Na druhou stranu se tato reprezentace příliš nehodí k úpravám barev, či definici barvy uživatelem. K těmto účelům vznikl například model HSV (a jemu podobné).

Existuje mnoho variant modelu RGB, které ho různými způsoby vylepšují a zpřesňují. Asi nejrozšířenější je sRGB který je standardem systému Windows. Používá se zejména proto, že odpovídá reálným možnostem většiny monitorů [9]. Dále například Adobe RGB vyvinutý firmou Adobe pro větší rozsah barev zejména v zeleno-modré, který však většina běžných monitorů již není schopna zobrazit, či Apple RGB, CIE RGB, ColorMatch RGB, atd. [11].



Obrázek 2.2 – Jednotková krychle modelu RGB (převzato z [10])

Normalizovaný RGB (rg model)

V normalizovaném RGB modelu popisují barevnou informaci složky *r* a *g*. Složka *b* je redundantní jelikož $r + g + b = 1$. Výpočet normalizovaného RGB popisuje rovnice (1). Díky normalizaci jsou složky *r* a *g* invariantní vůči změnám intenzity a stínování [8].

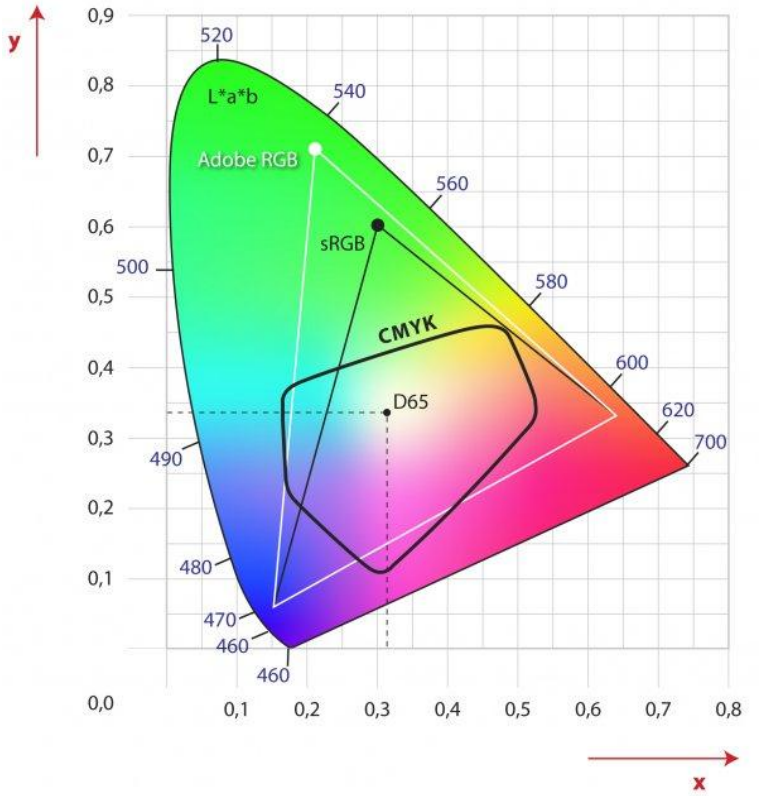
$$\begin{bmatrix} r \\ g \\ b \end{bmatrix} = \begin{bmatrix} R/(R+G+B) \\ G/(R+G+B) \\ B/(R+G+B) \end{bmatrix} \quad (1)$$

Tento barevný model využiji při konstrukci deskriptoru využívajícího kombinaci barevných histogramů a histogramů gradientů.

Model L*a*b (Lab CIELAB)

Tento model definovaný v CIE 1976 dokáže obsáhnout celé viditelné spektrum a je nezávislý na použitém zařízení. Je proto často používán jako referenční.

Jako většina modelů obsahuje tři složky jednu jasovou – *L* (Luminance) a dvě barvonosné – *a*, *b*. Tyto složky popisují barvu v určité ose. Složka *a* popisuje barvu bodu ve směru od zeleno-modré po červeno-purpurovou. Složka *b* ve směru od modro-purpurové po zeleno-žluto-červenou [9].



Obrázek 2.3 – Barevný model L*a*b v porovnání s dalšími modely (převzato z [20])

Oponentní modely

Existuje mnoho oponentních modelů. Všechny se ale zakládají na stejném principu. Obsahují 3 složky. Jednu jasovou složku (luminance) a dvě barvonosné (chrominance), založené na teorii oponentních barev.

Tato teorie říká, že lidé nevnímají červeno-zelené a modro-žluté odstíny. Můžeme tedy zavést tři komponenty oponentního modelu [7]:

- jasový kanál – *O1*
- červeno-zelený kanál – *O2 = G − B*
- modro-žlutý kanál – *O3 = B − Y = B − (R + G)*

Důsledkem je například fakt, že žlutý text na bílém papíře, či modrý text na černém papíře je obtížné přečíst, zatímco černý text na bílém papíře je snadno čitelný.

Důležitá fakta vyplývající z teorie oponentních barev [7]:

- Odstín je reprezentován úhlem vektoru v prostoru (O2, O3).
- Sytost je reprezentována velikostí vektoru v prostoru (O2, O3).
- Jasová složka nese nejvíce informace, proto je například možné barvonosné kanály podvzorkovat a využít této skutečnosti při kompresi obrazu.

Tyto modely mohou být závislé na zařízení, či nikoliv. Zástupcem nezávislého modelu je například zmiňovaný L*a*b. Závislé na zařízení jsou například modely YCbCr, YIQ, či YCC.

Následující rovnice (2) ukazuje převod modelu RGB na YCC (LC_1C_2), kde *L* je jasová složka a *C1*, *C2* jsou barvonosné složky [12].

$$\begin{bmatrix} L \\ C_1 \\ C_2 \end{bmatrix} = \begin{bmatrix} 0{,}299R + 0{,}587G + 0{,}114B \\ 0{,}5(R+G) - B \\ 0{,}866(R-G) \end{bmatrix} \tag{2}$$

V článku [6] byl použit částečně normalizovaný oponentní model, využívající jednoduchého převodu z prostoru RGB. Jeho výpočet je tedy velice rychlý. Jeho složky značené *I, O_1, O_2* jsou vypočteny podle rovnice (3).

$$\begin{bmatrix} I \\ O_1 \\ O_2 \end{bmatrix} = \begin{bmatrix} (R+G+B)/3 \\ (R+G-2B)/4 + 0{,}5 \\ (R-2G+B)/4 + 0{,}5 \end{bmatrix} \tag{3}$$

Tento barevný model využiji také při konstrukci vlastního deskriptoru.

2.2 Gradienty

Jednou z důležitých vlastností obrazu (nejen z pohledu počítačového vidění) je informace o změnách v obraze. Tyto změny popisují takzvané gradienty. Gradient v určitém místě obrazu říká, jak se v tomto místě obraz mění. Gradient vektor, který má právě dvě složky (velikost a směr). Velikost gradientu udává, jak rychle se obraz mění (změna s velikým kontrastem, skoková změna) a směr udává orientaci této změny [15].

Gradienty se dají využít například k hledání hran v obraze, protože na hranách bývá velká změna intenzity, neboli velká hodnota velikosti gradientu.

Výpočet gradientu

Vektor gradientu se vypočítá jako kombinace parciálních derivací obrazu v horizontálním a vertikálním směru (viz rovnice (4)) [15].

$$\nabla I = \left(\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y}\right) \tag{4}$$

Výpočet jedné parciální derivace pro spojitou funkci *I* ve směru *x* [15]:

$$\frac{\partial I(x,y)}{\partial x} = \lim_{\Delta x \to \infty} \frac{I(x+\Delta x,y) - I(x,y)}{\Delta x} \tag{5}$$

Jelikož je ale obraz diskrétní funkce je třeba počítat derivaci jako aproximaci hodnot (obdobně i pro směr *y*) [15]:

$$\frac{\partial I(x,y)}{\partial x} = \frac{I(x+1,y) - I(x-1,y)}{2} \tag{6}$$

Sobelův operátor

Pro výpočet aproximací prvních derivací obrazu se používá například Sobelův operátor. Výpočet probíhá jako konvoluce obrazu *I* s jádrem *X* (horizontální směr) a *Y* (vertikální směr).

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} * I, \; G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} * I \tag{7}$$

Výsledná aproximovaná velikost gradientu je pak:

$$G = \sqrt{{G_x}^2 + {G_y}^2} \tag{8}$$

Směr gradientu se vypočítá podle následujícího vztahu:

$$\theta = \tan^{-1} \frac{G_y}{G_x} \tag{9}$$

2.3 Kódování informace

Abychom mohli obraz určitým, dostatečně jednoduchým, způsobem popsat, potřebujeme k tomu takzvaný deskriptor, což je v podstatě N-rozměrný vektor vypočítaný tak, aby pokud možno jednoznačně popsal obraz, či oblast v obraze. Podle toho se deskriptory dělí na globální a lokální.

Globální deskriptory popisují obraz jako celek. Oproti tomu lokální deskriptory popisují pouze určité regiony v obraze nalezené pomocí detektoru (body zájmu, klíčové body).

2.3.1 Globální popis

Z globálního popisu obrazu vyplývají výhody v podobě relativně jednoduchého popisu, tudíž bývá jejich výpočet rychlý. Na druhou stranu mohou selhávat například při různých transformacích obrazu. Takovou transformací může být změna měřítka, rotace obrazu, změna pohledu kamery, či změna světelných podmínek nebo expozice.

Jako triviální příklad je možné uvažovat za deskriptor průměrnou intenzitu obrazu. Je evidentní, že při změně expozice u jinak identických obrázků tento deskriptor selže.

Histogramy

Histogram je funkce počítající četnost vzorků spadajících do každé z disjunktních kategorií známých jako „biny“ (bins). Jednou možností grafické reprezentace histogramu je sloupcový graf, kdy výška každého sloupce určuje počet pozorování spadajících do příslušné kategorie.

Pokud *n* udává celkový počet vzorků a *k* je počet binů, potom histogram m_i splňuje následující podmínku:

$$n = \sum_{i=1}^{k} m_i \tag{10}$$

Neexistuje žádný „ideální“ počet binů. Různé počty a velikosti binů mohou popisovat rozdílné vlastnosti dat. Z toho důvodu je třeba tyto počty pečlivě volit v závislosti na konkrétním použití histogramu.

V počítačové grafice jsou histogramy hojně využívané. Například jako informativní nástroj při pořizování digitálních fotografií (živý histogram), či při úpravě fotografií. Nebo jako nástroj pro různé úpravy obrazu, jako je například ekvalizace histogramu.

Významné využití mají histogramy z pohledu počítačového vidění. Dají se využít jako základ deskriptorů pro popis obrazu. Využívají se pro globální i lokální popis.

Využití histogramů pro globální popis obrazu

- **Scalable Color:** Tento deskriptor využívá barevný histogram v prostoru HSV zakódovaný pomocí Haarovy transformace [13].
- **Color-Structure Descriptor:** Zachycuje jak barevný obsah, tak informace o jeho struktuře. Uvažuje obraz po segmentech 8x8 pixelů místo uvažování každého pixelu zvlášť jako u „Scalable Color“. Na rozdíl od barevného histogramu dokáže rozlišit obrázky, v nichž je určitá barva zastoupena ve stejném množství, ale v jiném rozložení [13].
- **Edge histogram:** Histogram hran reprezentuje prostorové rozložení pěti typů hran. Z důvodu důležitosti hran ve vnímání obrazu dokáže získat obrázky s podobným sémantickým významem. Jeho kvalita může být významně vylepšena kombinací s jiným deskriptorem, jako je například histogram [13].

- **Histogram orientovaných gradientů (HOG):** Obraz se rozdělí na malé prostorové části – buňky (cells). Pro každou takovou buňku se spočítá histogram směrů gradientů nebo orientací hran ze všech pixelů buňky. Pro zlepšení invariance vůči osvětlení se používá normalizace kontrastu. To se provádí pomocí akumulace energií lokálních histogramů v rámci větších prostorových regionů – bloků (blocks) a využití výsledku k normalizaci všech buněk v bloku [16].

2.3.2 Lokální popis

Lokální deskriptory se nesnaží popsat obraz jako celek, ale jako množinu „zajímavých“ oblastí. Tyto oblasti je nejprve potřeba nalézt (většinou lokální extrémy ve formě rohů, či hran). K vyhledání těchto oblastí, neboli klíčových bodů se používají takzvané detektory.

Nalezené oblasti je poté třeba opět vhodným způsobem popsat. Mezi nejznámější a nejpopulárnější lokální deskriptory patří SIFT (Scale-Invariant Feature Transform) a SURF (Speeded Up Robust Features).

SIFT (Scale Invariant Feature Transform)

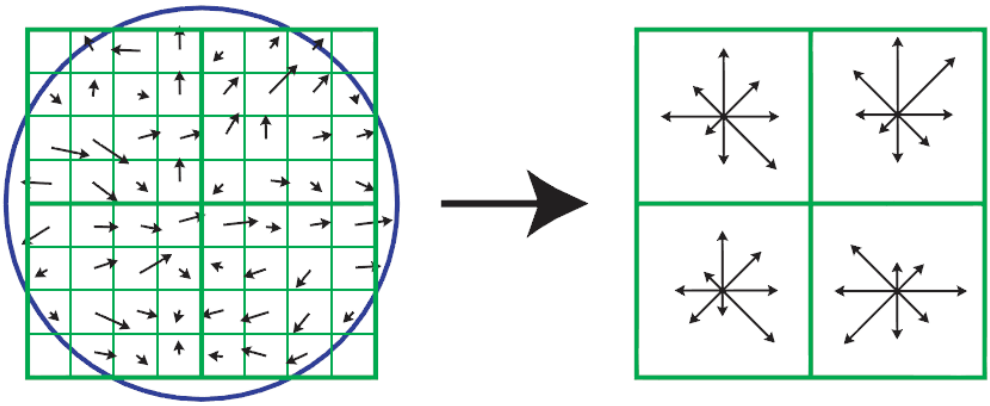
Tento deskriptor je, jak název napovídá, invariantní vůči změně měřítka, navíc i vůči rotaci, změně osvětlení, či kontrastu. Navíc je částečně invariantní vůči změně úhlu pohledu.

Před výpočtem SIFT deskriptorů je třeba předzpracování obrazu a vyhledání klíčových bodů. Jeden způsob nalezení klíčových bodů byl navržen v článku [14] v rámci návrhu SIFT deskriptoru. Probíhá v několika krocích.

Pro zachování Invariance vůči měřítku je nejprve třeba sestavit takzvaný „škálovatelný prostor“ (scale space). Díky němu je možné pracovat v různých úrovních měřítka. K tomuto účelu je potřeba odstranit detaily, což se provádí pomocí gaussovského rozmazání (neboli konvolucí s Gaussovou funkcí). Podle [14] je použito 5 úrovní rozmazání. Tím vnikne jedna takzvaná oktáva. Poté se obraz zmenší na polovinu a proces rozmazání se opakuje. Tím se získá druhá oktáva. Opět podle [14] jsou použity 4 oktávy. Soubor těchto oktáv tvoří potřebný „scale space“. V několika dalších, relativně náročných, krocích jsou vypočteny klíčové body obrazu.

Pro výsledné klíčové body jsou poté vypočítány SIFT deskriptory. Kolem klíčového bodu se vybere okolí 16x16 pixelů. To je rozděleno na 16 oblastí o velikosti 4x4. Poté je pro každý pixel vypočítána velikost a směr gradientu (relativně vzhledem k orientaci klíčového bodu). Dále jsou váhovány Gaussovou funkcí (viz Obrázek 2.4). V každé z 16ti oblastí jsou poté gradienty sumarizovány do jednoho histogramu o osmi sloupcích (neboli směry gradientů jsou kvantizovány do osmi směrů). Pro zachování invariance vůči osvětlení jsou histogramy normalizovány [14].

Výsledkem je tedy 16 histogramů gradientů o osmi hodnotách orientace, neboli 128 rozměrný vektor pro každý klíčový bod v obraze.



Obrázek 2.4 – Ilustrace SIFT deskriptoru pro velikost okolí 8x8 [14]. Váhování Gaussovou funkcí naznačuje modrá kružnice.

Jak je vidět, výpočet SIFT deskriptorů je výpočetně relativně náročný vzhledem k velkému množství zpracování obrazu. Právě rychlost a jednoduchost výpočtu je jednou z hlavních výhod globálních deskriptorů.

3 Vyhledávání (Image Retrieval)

Předchozí kapitola se mimo jiné zabývala různými metodami popisu obrazu. Výsledkem každé takové metody je deskriptor, což je obecně *N*-rozměrný vektor, který určitým způsobem kóduje informaci o obraze. Aby bylo možné deskriptory využít pro vyhledávání podobných obrázků, je třeba umět tyto deskriptory vhodně porovnávat a dokázat efektivně vyhledávat podobné deskriptory. Neboli vyhledávat vektory, které leží blízko u sebe v *N*-rozměrném prostoru. Tímto problémem se pak zabývají takzvané indexační metody.

3.1 Porovnávání deskriptorů

Pro dva body v *N*-rozměrném prostoru $x = (x_1, x_2, ..., x_N)$ a $y = (y_1, y_2, ... y_N)$ je možné definovat mnoho různých vzdáleností s různým zaměřením a způsobem použití. Popisuji zde jen několik z nich.

L_1 vzdálenost (Manhattanská vzdálenost)

Nejjednodušší způsob výpočtu vzdálenosti dvou bodů. V dvourozměrném prostoru představuje vzdálenost měřenou po odvěsnách pravoúhlého trojúhelníka. Pro *N*-rozměrný prostor je definována jako:

$$d_{L1}(x, y) = \sum_{i=1}^{N} |x_i - y_i| \tag{12}$$

Používá se pro ordinální proměnné, jež lze seřadit podle velikosti.

L_2 vzdálenost (Euklidovská vzdálenost)

Asi nejznámější a nejpoužívanější vzdálenost. V dvourozměrném prostoru je definována jako délka úsečky spojující dva body (nebo také délka přepony pravoúhlého trojúhelníka). Pro *N*-rozměrný prostor je definována jako:

$$d_{L2}(x, y) = \sqrt{\sum_{i=1}^{N} (x_i - y_i)^2} \tag{13}$$

Minkowského metrika

Obecná forma výpočtu vzdálenosti. Podle parametru r může odpovídat např. Euklidovské vzdálenosti ($r = 2$). Vyšší hodnoty parametru r zvětšují významnost rozdílů [18].

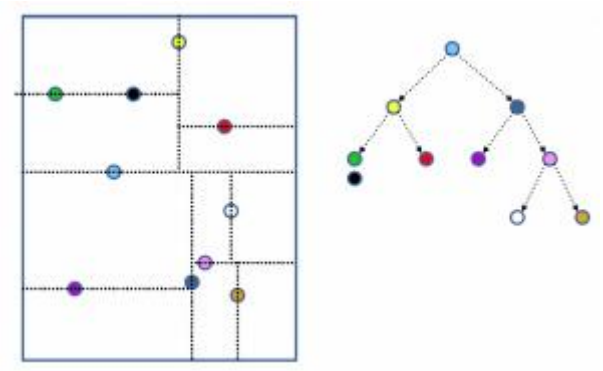
$$d_M(x, y) = \sqrt[r]{\sum_{i=1}^{N}(x_i - y_i)^r} \tag{14}$$

Pro účely této práce jsou nedůležitější vzdálenost L_1 a L_2.

3.2 Indexační metody

3.2.1 Kd-tree

Kd-tree je binární strom ve kterém každý uzel představuje bod v *k*-dimenzionálním prostoru. Každý uzel, který není listem, rozděluje prostor na dvě části (podprostory). Každému uzlu je přidělena jedna z os prostoru k níž je kolmá hyperplocha rozdělující prostor.

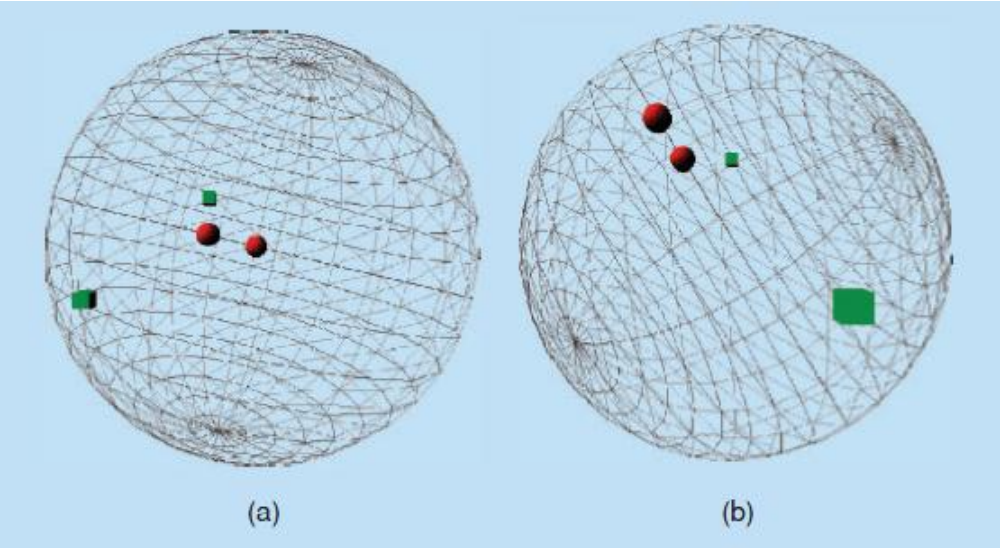


Obrázek 3.1 – Kd-tree (převzato z [19])

Příliš se nehodí pro vyhledávání blízkých sousedů v prostorech s vysokou dimenzí. Pokud je prostor *k*-rozměrný, potom by počet bodů *N*, měl být $N >> 2^k$, aby použití kd-tree bylo efektivní.

3.2.2 LSH

LSH (Locality Sensitive Hashing) je založeno na jednoduché myšlence, že pokud jsou dva body v prostoru blízko sebe, potom po „projekci“ do prostoru s nižší dimenzí tyto body blízko sebe zůstanou (viz Obrázek 3.2). Dva body, které jsou v prostoru blízko u sebe (znázorněné jako červené koule), zůstanou u sebe i po projekci do dvourozměrného prostoru. Zároveň je jedno v jakém směru je tato projekce provedena. Na druhou stranu je třeba si uvědomit, že pokud jsou dva body daleko od sebe (znázorněné jako zelené krychle), mohou být po projekci blízko, či daleko od sebe. A to v závislosti na směru, ze kterého je projekce provedena [4].



Obrázek 3.2 – Příklad projekce bodů v prostoru (převzato z [4])

Základem metody LSH je skalární projekce (dot product) definovaný jako $h(\vec{v}) = \vec{v} \cdot \vec{x}$ kde $\vec{v}$ je bod v *N*-rozměrném prostoru a $\vec{x}$ je vektor s prvky vybranými náhodně podle Gaussovské distribuce například v rozsahu *(0..1)*. Tato skalární projekce je kvantizována do takzvaných hashovacích košů (buckets) tak, aby se blízké body v originálním *N*-rozměrném prostoru po projekci ocitly ve stejném koši. Výsledná hashovací funkce je pak:

$$h^{x,b}(\vec{v}) = \left\lfloor \frac{\vec{x} \cdot \vec{v} + b}{w} \right\rfloor \tag{15}$$

kde *w* je šířka každého kvantizačního koše, *b* je náhodné číslo vybrané z uniformního rozložení od 0 do *w* a ⌊ . ⌋ je operace zaokrouhlení dolů (floor operation) [3]. Pro výslednou hashovací funkci je samozřejmě číslo *b* a vektor $\vec{v}$ fixní [5].

Hashovací funkce dále musí splňovat tu vlastnost, že pokud jsou dva body v prostoru blízko u sebe, musejí s vysokou pravděpodobností po projekci „padnout“ do stejného koše (bucket). Neboli pro

dva body *p* a *q* v *N*-rozměrném prostoru, které jsou blízko u sebe je veliká pravděpodobnost P_1, že se ocitnou ve stejném koši:

$$P_H[h(p) = h(q)] \geq P_1 \; pro \; \|p - q\| \leq R_1 \tag{16}$$

A zároveň pro každé dva body *p* a *q*, které jsou v *N*-rozměrném prostoru daleko od sebe je malá pravděpodobnost $P_2 < P_1$, že se ocitnou ve stejném koši:

$$P_H[h(p) = h(q)] \leq P_2 \; pro \; \|p - q\| \geq cR_1 = R_2 \tag{17}$$

přičemž $R_2 > R_1$ a $\|.\|$ je L_2 vzdálenost vektorů. Díky linearitě skalární projekce je rozložení velikosti vzdáleností dvou bodů $\|h(p) - h(q)\|$ proporcionální vůči $\|p - q\|$ a tudíž $P_1 > P_2$ [2].
Rozdíl pravděpodobností P_1 a P_2 je dále možné zvětšit provedením k projekcí, jelikož $(\mathrm{P}_1/\mathrm{P}_2)^{\mathrm{k}} > (\mathrm{P}_1/\mathrm{P}_2)$. Výsledná projekce je tedy získána provedením *k* nezávislých skalárních projekcí $h^{x,b}(\vec{v})$. Myšlenka je taková, že podobné (blízké v prostoru) body budou zahashovány do stejného koše ve všech projekcích.

Zvětšení šířky koše *w* způsobí zvýšení počtu bodů, které se v něm objeví. Pro finální získání nejbližšího bodu je potom třeba provést lineární prohledání potenciálních sousedů. Je zřejmé, že změna velikosti *w* je otázkou rovnováhy mezi velikostí hashovací tabulky a počtem potenciálních sousedů, mezi kterými musíme provést lineární prohledání [4]. Optimální hodnota parametru *w* závisí na aplikaci a datech, ale dle experimentálních poznatků uvedených v [17] poskytuje dobré výsledky hodnota $w = 4$.

4 Návrh řešení

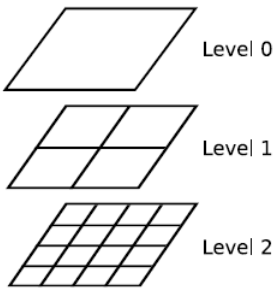
V této kapitole nejprve popisuji deskriptor navržený v článku [6], který budu dále používat jako referenční. Poté následuje návrh vlastního deskriptoru a jeho dvou modifikací.

4.1 Referenční deskriptor

Vlastní návrh deskriptoru jsem se rozhodl založit na článku „Scalable Near Identical Image and Shot Detection“ [6]. V tomto článku je navržen globální deskriptor popisující obraz pomocí barevných histogramů. Tento deskriptor je velice dobře navržený a poskytuje dobré výsledky při vyhledávání velmi podobných obrázků, jak popisuje zmíněný článek [6]. Proto ho budu používat při porovnávání výsledků vlastního návrhu.

Deskriptor navržený v tomto článku využívá oponentního barevného modelu (IO_1O_2), který je podrobněji popsán v části 2.1.

Deskriptor dále využívá rozdělení obrazu do takzvané prostorové pyramidy o třech úrovních. První úroveň (Level 0) popisuje celý obraz, ve druhé je obraz rozdělen na čtyři části a ve třetí na 16 částí. Každá část je popsána histogramem o určitém počtu binů. Každý bin je normalizován na velikost 1Byte. Pro každou úroveň pyramidy je použito 128 B. Obrázek 4.1 ukazuje způsob dělení obrazu do pyramidy a počty binů pro histogramy jednotlivých barevných složek. V této konfiguraci má výsledný deskriptor vzniklý konkatenací těchto histogramů velikost *(64 + 32 + 32) + 4(16 + 8 + 8) + 16(4 + 2 + 2) = 384 B*.



Level	I	O_1	O_2
0	64	32	32
1	16	8	8
2	4	2	2

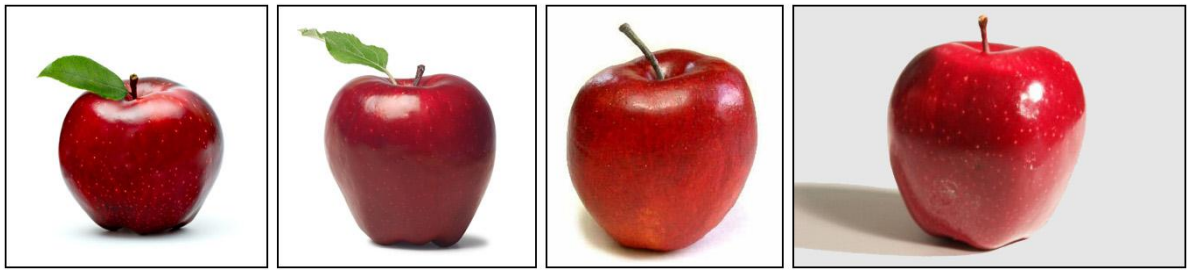
Obrázek 4.1 – Dělení úrovní obrazu do prostorové pyramidy (vlevo) a velikost dat přidělených pro jednotlivé úrovně a barevné složky (vpravo) [6]

Výhoda tohoto přístupu spočívá v rychlosti výpočtu deskriptoru a v jeho malé velikosti. Při výpočtu stačí akumulovat body do histogramů na nejnižší úrovni (Level 2) a vyšší úrovně vypočítat součtem těchto histogramů. Až poté se provede normalizace na 1 B pro bin.

Tento deskriptor použiji jako referenční při testování vlastních deskriptorů. Dále v práci ho proto budu označovat jako „**Referenční deskriptor**“.

4.2 Návrh deskriptoru

Při návrhu vlastního deskriptoru jsem vycházel z poznatků popsaných v teoretické části práce a cílem bylo navrhnout deskriptor, který bude mít vhodné vlastnosti pro vyhledávání podobných obrázků a ověřit, jak se tyto vlastnosti liší od vlastností Referenčního deskriptoru. Tento deskriptor by měl být schopen obrázky dostatečně rozlišovat, tzn. poznat, zda jsou obrázky opravdu podobné, ale zároveň by byla vhodná určitá míra tolerance k některým druhům zkreslení, či změnám v obraze, při nichž lze obrázek stále považovat za podobný, či dokonce stejný. Obrázek 4.2 ukazuje 4 obrázky, které je možné považovat za velice podobné (na všech je červené jablko). Čistě formálně se však jedná o rozdílné obrázky. Každý zobrazuje jiný objekt, jinak nasvícený, každý obrázek má jiné rozměry, či dokonce jiné rozlišení.



Obrázek 4.2 – "Podobné" obrázky. Jsou všechny opravdu podobné?

Lze tedy říci, že obrázky jsou rozdílné, ale dle lidského vnímání jsou podobné. Proto je třeba aby deskriptor poskytoval určitou míru tolerance. Velmi důležité je ale nalézt rovnováhu mezi tolerancí a rozlišovací schopností deskriptoru. V případě vysoké tolerance bychom mohli dostat výsledky, které se již příliš liší od originálu (viz Obrázek 4.3).

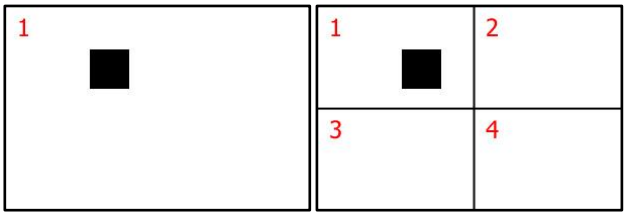
Obrázek 4.3 – Jsou toto obrázky stále podobné?

Vlastní návrh

Na základě těchto skutečností jsem navrhl deskriptor založený na čtyřech principech a využívající barevné histogramy. Používá vhodný barevný model, dělení obrazu na dlaždice, překryv dlaždic a váhování jednotlivých příspěvků do histogramů.

Při konstrukci deskriptoru jsem použil oponentní **barevný model** (IO_1O_2), který byl s úspěchem použit v Referenčním deskriptoru. **Dělení obrazu** je společně s vhodným barevným modelem krokem k vyšší rozlišovací schopnosti deskriptoru. Tento předpoklad vyplývá z faktu, že popíšeme-li určitým způsobem jednotlivé části obrazu, je celkový popis přesnější, než pokud popíšeme stejným způsobem obraz jako celek.

Pro demonstraci uvedu zjednodušený příklad ukazující tuto skutečnost. Pokud uvažujeme situaci na obrázku (viz Obrázek 4.4) a jako popis obrazu použijeme pouze konstatování, zda se na obrázku vyskytuje černý čtverec či ne, bude popis obrázku jako celku pouze [ANO]. Pokud ale obraz rozdělíme na 4 části bude popis [ANO, NE, NE, NE]. Je vidět, že takovýto popis je mnohem přesnější.

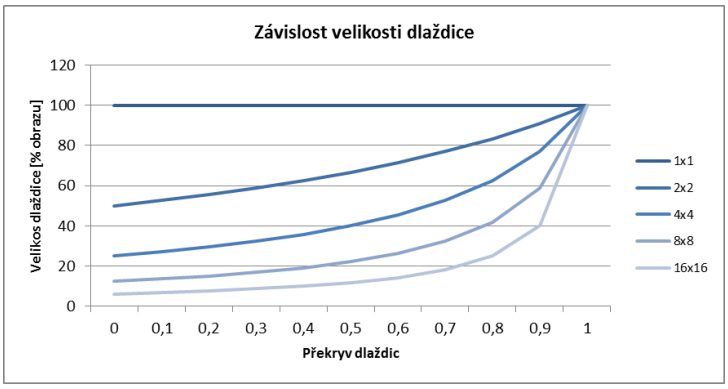


Obrázek 4.4 – Příklad dělení obrazu na dlaždice. Vlevo: celý obraz. Vpravo: obraz rozdělený na 4 části.

Je tedy zřejmé, že čím jemnější dělení obrazu, tím přesnější popis získáme. Samozřejmě to platí i při popisu pomocí barevných histogramů.

Posledním krokem směrem k rozlišovací schopnosti deskriptoru je množství informace použité k popisu obrazu, neboli velikost dat použitých pro deskriptor. Jako vhodnou velikost jsem se rozhodl použít **1024 B**, což je více než u Referenčního deskriptoru, ale stále je to relativně malá velikost. Počet dat použitých pro jednu dlaždici, neboli velikost jednotlivých histogramů, samozřejmě závisí na počtu dlaždic. Stejně jako v Referenčním deskriptoru použiji pro jasovou složku obrazu *I* **dvojnásobek** dat než pro jednotlivé chromatické složky O_1 a O_2.

Jak již bylo řečeno, deskriptor bude využívat překryvu jednotlivých dlaždic. **Překryv dlaždic** je na rozdíl od zvyšování počtu dlaždic krokem směrem k toleranci deskriptoru. Je možné tak usoudit například ze skutečnosti, že pokud by v extrémním případě byl překryv dlaždic 100 % *(1,0)* znamenalo by to v zásadě, že všechny dlaždice pokrývají celý obraz, neboli dělení obrazu se neprojeví. Závislost velikosti dlaždice (plochy pokryté jednou dlaždicí) na dělení obrazu a překryvu dlaždic ukazuje následující diagram (viz Obrázek 4.5).



Obrázek 4.5 – Závislost velikosti dlaždice na dělení obrazu a překryvu dlaždic

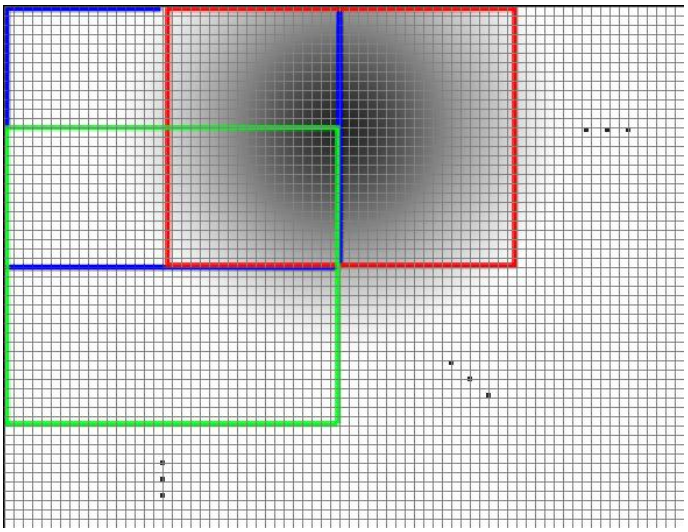
Pro zachování rovnováhy mezi tolerancí a rozlišovací schopností je třeba, aby překryv dlaždic byl zvolen s rozvahou. Z toho důvodu jsem provedl testy chování deskriptoru a na základě výsledků jsem poté navrhl optimální parametry dělení obrazu a překryvu dlaždic. Toto testování a výsledné parametry popisuji v kapitole 6.1.

Druhým způsobem zvýšení tolerance deskriptoru je **váhování** příspěvků histogramu pomocí 2D Gaussovy funkce. To znamená, že při vytváření histogramu, nepřispívá každý pixel hodnotou „1“, ale pouze hodnotou, která je rovna hodnotě Gaussovy funkce v daném místě obrazu (Gaussova funkce je normovaná na hodnotu 1,0).

Dvourozměrná Gaussova funkce je počítána podle rovnice (18), kde *A=1*, určuje maximální hodnotu Gaussovy funkce (amplitudu), x_0 a y_0 jsou souřadnice středu obrazu (středu aktuální dlaždice). Hodnoty σ_x a σ_y jsou vypočítány z velikosti dlaždice tak, že $\sigma_x = 0{,}3(\text{šířka}/2 - 1) + 0{,}8$ a analogicky pro σ_y.

$$f(x,y) = Ae^{-\left(\frac{(x-x_0)^2}{2\sigma_x^2}+\frac{(y-y_0)^2}{2\sigma_y^2}\right)} \tag{18}$$

Takovéto parametry zaručují, že výsledná Gaussova funkce bude ideálně pokrývat každou dlaždici obrazu. Obrázek 4.6 ilustruje umístění Gaussovy funkce pro jednu dlaždici (červené ohraničení). V tomto případě je hodnota 1,0 zakreslena černou barvou.



Obrázek 4.6 – Dělení obrazu na dlaždice s překryvem a váhování příspěvků pomocí Gaussovy funkce

Výsledný deskriptor tedy vznikne tak, že se pro každou dlaždici vypočítá histogram (s váhováním příspěvků) pro každý „barevný“ kanál zvlášť a provede se jeho normalizace tak, aby 1 bin histogramu zabíral 1 Byte. Následně se tyto histogramy konkatenují, čímž vznikne jeden vektor (o velikosti 1024 B).

Lze předpokládat, že takovýto deskriptor by měl být méně náchylný k lokálním změnám v obraze, jako je například malý posun objektu ve statickém obraze, či posun celého obrazu. Neměl by také být náchylný ke změnám na okrajích obrazu, jelikož díky váhování jednotlivých dlaždic bude okraj přispívat celkovému deskriptoru malou měrou.

Navržený deskriptor budu označovat jako „**Barevný deskriptor**“. Měl by být vhodný k vyhledávání „velmi podobných“ obrázků. Nepředpokládám příliš velkou invarianci vůči změně pohledu, či rotaci obrazu jako například u SIFT. Mohl by ale v určitých situacích poskytovat lepší výsledky než deskriptor navržený v článku [6] (Referenční deskriptor) a stále být méně náročný na výpočet, než lokální metody (např. SIFT).

Jako alternativu jsem dále navrhl dva deskriptory využívající k popisu obrazu histogramy gradientů. Dělení obrazu, překryv dlaždic i váhování zůstává stejné, jako u Barevného deskriptoru, ale v prvním případě jsou použity histogramy velikostí gradientů. Pro každou dlaždici je vypočítán pouze jeden histogram. Výpočet velikosti gradientů se provádí pomocí Sobelova operátoru. Tento deskriptor je dále označován jako „**Gradient deskriptor**“. Záměrně zde nepočítám se směry gradientů a budu testovat, jak se takovýto deskriptor chová.
Poslední deskriptor, který dále označuji jako „**Gradient+RG deskriptor**“, je jakousi kombinací předchozích dvou. Pro každou dlaždici obrazu jsou opět vypočítány tři histogramy. Jeden histogram velikostí gradientů a dva barevné histogramy. V tomto případě je ale použit normalizovaný model RGB (kapitola 2.1, rovnice (1)). Jak napovídá název deskriptoru, jsou použity pouze složky R a G (normalizovaný červený a zelený kanál).

Opět bude předmětem testování, jak tento deskriptor pracuje, a k jakému účelu je možné ho použít.

5 Implementace

V první fázi jsem implementoval výpočet Referenčního deskriptoru, tak jak byl navržen v článku [6], jelikož ho při testování využiji pro srovnání výsledků.

Dále jsem implementoval funkce pro výpočet deskriptoru podle vlastního návrhu (Barevný deskriptor). Pro účely následného testování je implementace provedena tak, aby bylo možné snadno měnit parametry deskriptoru (dělení obrazu, překryv dlaždic a velikost jednotlivých histogramů).

Výpočet deskriptoru pracuje tak, že nejdříve je obrázek převeden do příslušného barevného modelu (IO_1O_2) a podle velikosti obrázku a podle parametrů se předpočítají hodnoty 2D Gaussovy funkce do matice o velikosti rovnající se velikosti jedné dlaždice. Hodnoty Gaussovy funkce jsou vypočítány s využitím funkce knihovny OpenCV pro výpočet jednorozměrné Gaussovy funkce. Tyto funkce jsou vypočítány 2 (pro každý rozměr dlaždice). Z těchto dvou vektorů je následně vypočítána matice 2D Gaussovy funkce.

Poté jsou vypočítány histogramy pro všechny dlaždice s váhováním podle hodnot této Gaussovy funkce. Histogramy jsou poté normalizovány podle velikosti dlaždice a výsledné histogramy jsou konkatenovány do jednoho vektoru. Tento vektor je vypočítaný deskriptor.

Výpočet dalších dvou deskriptorů (Gradient a Gradient+RG deskriptor) pracuje stejně jako výpočet Barevného deskriptoru. Používají ale jiné předzpracování obrazu. Pomocí Sobelova operátoru (použita implementace v OpenCV) se nejdříve vypočítají velikosti gradientů. Tento výsledek je použit pro výpočet Gradient deskriptoru.

Pro výpočet Gradient+RG deskriptoru je třeba navíc převést originální obrázek do prostoru normalizovaného RGB. Oba výsledky jsou poté použity pro výpočet histogramů s tím, že místo modrého kanálu (B) jsou použity hodnoty gradientů (Gr).

Všechny deskriptory jsou samozřejmě implementovány se stejným rozhraním. Detaily implementace viz Příloha 1.

Je zde třeba poznamenat, že cílem práce není co nejefektivnější implementace výpočtu deskriptorů, ale navržení a otestování metody. Proto jsem se při implementaci výpočtu deskriptorů zaměřil hlavně na modifikovatelnost (aby bylo možné v průběhu návrhu a testování měnit součásti výpočtu a parametry) a přehlednost implementace. Přesto jsem se ale snažil, aby byla rychlost výpočtu na použitelné úrovni. Příklad rychlosti výpočtu deskriptorů a vytvoření indexu ukazuje následující tabulka (viz Tabulka 5.1).

Jak je vidět, tak výpočet Barevného deskriptoru je o cca 36 % pomalejší, než výpočet Referenčního deskriptoru. To není nijak překvapivé vzhledem k faktu, že výpočet tohoto deskriptoru je složitější a výsledný deskriptor více než 2krát větší (viz návrh deskriptoru).

Výpočet Gradient+RG deskriptoru je dokonce o 50 % pomalejší, než výpočet Referenčního deskriptoru, což vyplývá z faktu, že obrázek je třeba nejdříve 2x předzpracovat (převod do normalizovaného RGB a navíc výpočet gradientů).

Deskriptor	Doba výpočtu deskriptorů [s]		
	5000 obrázků(76,8 kpx)	2125 obrázků (0,44 Mpx)	10200 obrázků (0,3 Mpx)
Referenční	20	48	170
Barevný	31	76	263
Gradient	29	67	235
Gradient+RG	41	94	327

Tabulka 5.1 – Rychlost výpočtu deskriptorů pro různé počty obrázků o různých rozlišeních. Výpočty probíhaly na procesoru Intel Core i7 – 3,6Ghz.

Ve druhé fázi jsem implementoval experimentální aplikaci, která slouží pro testování a demonstraci fungování navržených deskriptorů a výsledné metody vyhledání podobných obrázků.

Součástí této aplikace jsou funkce pro vypočítání deskriptorů pro skupinu obrázků (ve složce a podsložkách) nebo pro jednotlivé snímky videa a vytvoření indexu (LSH) těchto deskriptorů pro účely vyhledávání. K tomu účelu používám implementaci LSH obsaženou v knihovně OpenCV. Na základě tohoto indexu je poté možné vyhledávat podobné obrázky (vstupem je jeden obrázek a výstupem je předem daný počet „podobných“ obrázků), což realizuje další implementovaná funkce. Pomocí dalších funkcí je vypočítaný index, spolu s nezbytnými parametry uložen do souborů.

Aplikace dále obsahuje funkce, které s využitím tohoto indexu, nebo na základě vypočítaných deskriptorů (metodou hrubé síly) porovnají dvě videa a naleznou podobné, či stejné segmenty v těchto videích. Poslední možností je vyhledání podobných obrázků (snímků) ve videu na základě jednoho obrázku.

Popis funkcí a fungování aplikace viz Příloha 1.

6 Experimenty a testování

V této kapitole se zabývám daty vytvořenými pro účely testování, popisem a provedením samotných testů a vyhodnocením jejich výsledků.

V první fázi, kterou se zabývá kapitola 6.1, budu testovat vlastnosti samotných deskriptorů při různých nastaveních parametrů. Pro tyto účely jsem vytvořil vlastní sadu testovacích obrázků, kterou popisuji v části 6.1.1. Tyto obrázky jsou určeny pro testování chování deskriptorů při určitých změnách obrazu.

Na základě výsledků těchto testů poté navrhnu vhodné parametry deskriptorů a budu je dále testovat pomocí druhé datové sady. Ta obsahuje videa devět videí, a jejich kopie s různými úpravami. Tato videa popisuji v části 6.1.3. Rozhodl jsem se s výhodou použít videa, která jsou v podstatě souborem obrázků.

V druhé fázi budu testovat použitelnost deskriptorů v aplikaci pro vyhledávání podobných obrázků. Testování vyhledávání podobných obrázků proběhlo opět ve videích. Pro tyto účely jsem vytvořil další sadu videí, ale jako její základ jsem použil videa použitá pro testování deskriptorů. Popisem datové sady pro testování vyhledávání, popisem testů, samotným testováním a vyhodnocením zjištěných výsledků se zabývám v kapitole 6.2.

6.1 Testování deskriptorů

V této kapitole popisuji dvě datové sady použité pro testování deskriptorů. První sada je složena z obrázků a druhá z videí. Dále zde popisuji provedené testy s využitím těchto datových sad a vyhodnocení výsledků.

Výsledkem je navržení vhodných parametrů deskriptorů a zhodnocení jejich vlastností při těchto parametrech.

6.1.1 Datová sada pro testování vlivu parametrů deskriptorů

První datová sada je určená pro testování, jak se deskriptory chovají při různých změnách v obraze podobně jako v [1]. Obsahuje 7 skupin obrázků. Každá skupina reprezentuje jeden typ zkreslení (změny) a obsahuje 5 obrázků s různou úrovní tohoto zkreslení. Tato zkreslení jsou – **změna velikosti obrazu, posunutí obrazu, rozostření, vliv JPEG komprese, změny expozice, vliv šumu a vinětace**. Každá skupina obsahuje referenční obrázek (originál) ke kterému budou vztaženy výsledky jednotlivých testů.

Změna velikosti obrazu

Tato skupina testuje, jaký vliv na chování deskriptoru má změna velikosti obrazu (přesamplování). Jelikož se jedná o globální deskriptory, měly by být vůči této změně teoreticky invariantní. Obrázky této skupiny mají velikost 80, 60, 40 a 20 % originálu.

Obrázek 6.1 – Skupina obrázků se změnou velikosti

Posunutí obrazu

Slouží pro testování, jak se deskriptory chovají při posunu obrazu. Teoreticky by navržené deskriptory měly být k menšímu posuvu obrazu tolerantní. Je použit horizontální posuv o 2,5, 5, 7,5 a 10 % obrazu.

Obrázek 6.2 – Obrázky s rostoucím horizontální posuvem

Záměrně jsem zvolil obrázek, který je sice odlišný od obrázků v ostatních testech, ale lépe simuluje situaci, kdy se posouvá objekt v obraze, ale přitom celkový obraz zůstává velice podobný.

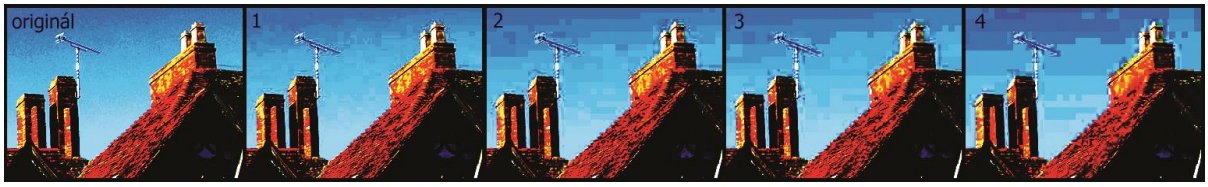
Rozostření obrazu

Z podstaty navržených deskriptorů teoreticky vyplývá, že by určitá hodnota rozostření neměla mít příliš velký vliv a tudíž by deskriptory měly být vůči rozostření invariantní. Tento předpoklad testuje tato skupina obrázků. Použil jsem gaussovské rozostření s poloměrem 2, 4, 6 a 8 pixelů.

Obrázek 6.3 – Obrázky s různou hodnotou rozostření

JPEG komprese

Testování vlivu JPEG komprese. Program, který byl použit pro uložení obrázků definuje kvalitu na stupnici 12..0, kde 12 je nevyšší kvalita. S kvalitou 12 byl uložen referenční obrázek. Další obrázky byly uloženy ve kvalitě 8, 4, 2 a 0. Na obrázku 5.4 jsou pro ilustraci pouze výřezy originálních obrázků.



Obrázek 6.4 – Ilustrace použité JPEG komprese. Obrázky jsou upraveny pro zvýraznění artefaktů.

Změna expozice

Pro testování, jak se deskriptory chovají při této relativně zásadní změně v obraze. Použil jsem změnu expozice v hodnotách -4, -2, 0, +1, a +2EV. Obrázek s hodnotou -4EV (nejtmavší) je zde považován za referenční.

Obrázek 6.5 – Obrázky s různou hodnotou expozice

Šum

Obrázky pro testování vlivu šumu. Jedná se o monochromatický šum s uniformním rozložením. Šum byl do obrázků přidán uměle s hodnotami intenzity 20, 40, 60 a 80 %.

Obrázek 6.6 – Obrázky z různou úrovní šumu

Vinětace

Na závěr testuji vliv vinětace. Z konstrukce deskriptorů (váhování gaussovou funkcí) vyplývá, že by měly být vůči vinětaci invariantní. Jelikož je ale vinětace těžko „měřitelná“, postačí pro určení „míry“ vinětace jednotlivých obrázků ilustrace na obrázku 5.7.

Obrázek 6.7 – Obrázky ukazující narůstající hodnotu vinětace

6.1.2 Testování vlivu parametrů deskriptorů

Pro testování vlivu parametrů na navržené deskriptory jsem použil datovou sadu popsanou v předchozí části. Výsledkem testů jsou diagramy ukazující míru odlišnosti vypočítaných deskriptorů jednotlivých obrázků od deskriptoru referenčního obrázku (obrázek beze změny). Jako metrika je použita **L_2 vzdálenost**.

Tyto testy jsem provedl pro různé nastavení parametrů deskriptorů (počet dlaždic, překryv dlaždic). Velikost deskriptorů zůstává konstantní – 1024 B pro mé deskriptory a 384 B pro referenční deskriptor.

Vzhledem k zaměření práce na využití barevných histogramů a s přihlédnutím k velkému množství výsledků se zde zaměřím na vyhodnocení výsledků pro „Barevný deskriptor“. Výsledky pro další dva deskriptory viz Příloha 2.

Pro každý obrázek v sadě byl vypočítán deskriptor a tyto deskriptory byly porovnány s deskriptorem referenčního obrázku. Výsledkem porovnání je L_2 vzdálenost těchto deskriptorů. Čím větší je tato vzdálenost, tím odlišnější jsou dva obrázky pro daný deskriptor.

Testy dělení obrazu na 2x2, 4x4, 8x8, a 16x16 dlaždic jsem provedl při konstantním překryvu dlaždic (0,5). Jelikož ale při dělení na 16x16 připadají na každou dlaždici pouze 4 B (na všechny 3 barevné kanály – 2+1+1), což je velice málo, přidal jsem do testování deskriptor dělený na 16x16 dlaždic s velikostí 8192 B (8 kB), kdy na každou dlaždici připadá 32 B. Toto nastavení je v diagramech reprezentováno označením 16x16_8k.

Testy překryvu dlaždic jsou provedeny při konstantním dělení na 4x4 dlaždice pro překryv 0 (bez překryvu), 0,2, 0,4, 0,6, 0,8 a 0,9.

Do testování je pro srovnání zahrnut Referenční deskriptor, což je deskriptor navrhnutý v článku [6], který jsem popsal v kapitole 4, jehož parametry zůstávají konstantní. V diagramech je reprezentován plnou čarou a označením „Ref“.

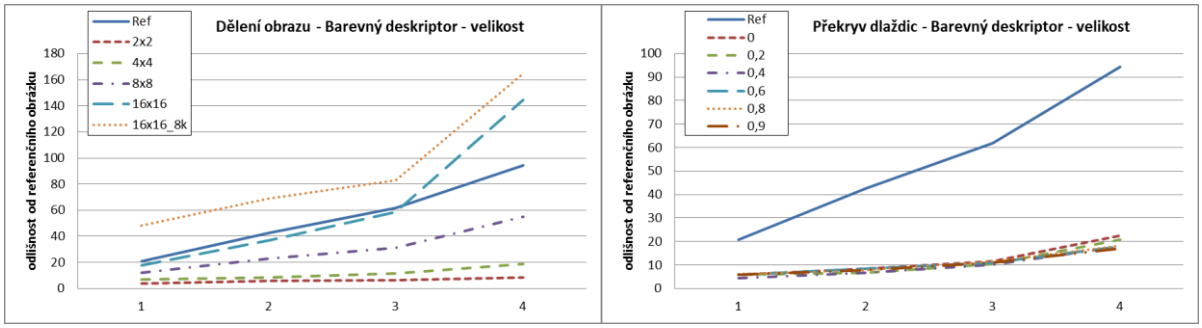
Dále je třeba zdůraznit, že ve výsledcích testů nejsou důležité absolutní hodnoty odlišnosti od referenčního obrázku. To plyne z faktu, že mé deskriptory jsou konstruovány jinak, než referenční deskriptor a zároveň mají jinou velikost. Proto je zde zkoumanou skutečností vývoj odlišnosti při rostoucí úrovni změny obrazu. „Konstantní“ hodnota odlišnosti ukazuje na tolerantnost deskriptoru ke konkrétní změně. Naopak čím strmější nárůst, tím více diskriminující je deskriptor k této změně.

Výsledky testování a vyhodnocení

Test změny velikosti obrazu

Výsledky ukazují, že čím jemnější dělení obrazu, tím je deskriptor náchylnější na změnu velikosti obrazu. Jelikož deskriptor bere v potaz obraz jako celek, měl by být k pouhé změně velikosti invariantní. To ale platí, pouze do určité míry. Při velkém zmenšení obrazu dochází vlivem převzorkování ke ztrátě dat a tudíž je zmenšený obraz „odlišný“. Proto je vhodné, aby byl deskriptor k malé změně velikosti tolerantní, ale zároveň byl schopný rozpoznat větší změny. To zde platí pro dělení na 4x4, či 8x8 dlaždic. Dělení na 2x2 je již příliš tolerantní.

Druhý diagram ukazuje, že změna překryvu dlaždic má jen zanedbatelný vliv na chování deskriptoru v této situaci.

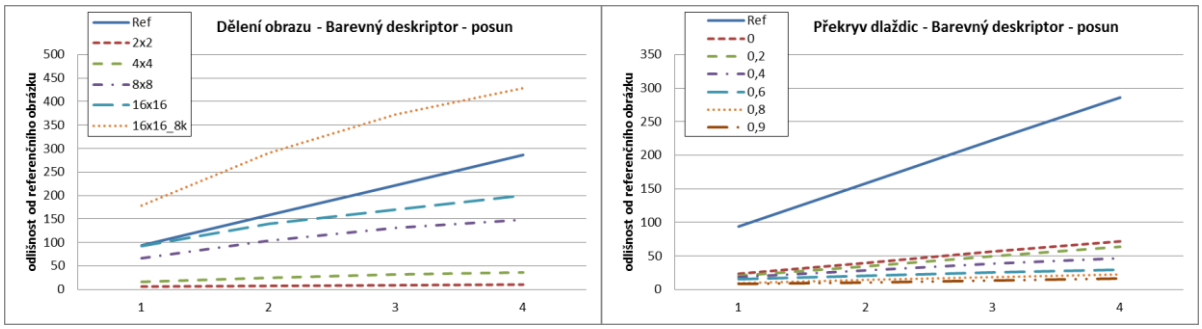


Obrázek 6.8 – Chování deskriptoru při změně velikosti obrazu pro různé dělení obrazu (vlevo) a pro různý překryv dlaždic (vpravo)

Test posunu obrazu

V této situaci jsem předpokládal, že při jemnějším dělení obrazu deskriptor lépe rozpozná změnu obrazu. Výsledky tohoto testu předpoklad potvrzují. Jelikož byl tento test navržen tak, aby zjistil, při kterém nastavení parametrů je deskriptor dostatečně tolerantní k posuvu obrazu, vychází jako nejvhodnější dělení na 4x4 dlaždic (2x2 je příliš tolerantní a 8x8 je již příliš diskriminující).

Z testu vlivu překryvu dlaždic vyplývá, že čím větší překryv dlaždic, tím tolerantnější je deskriptor k posunu obrazu. Jako kompromis mezi tolerantností a rozlišovací schopností deskriptoru zde vychází jako optimum hodnota překryvu dlaždic 0,4.

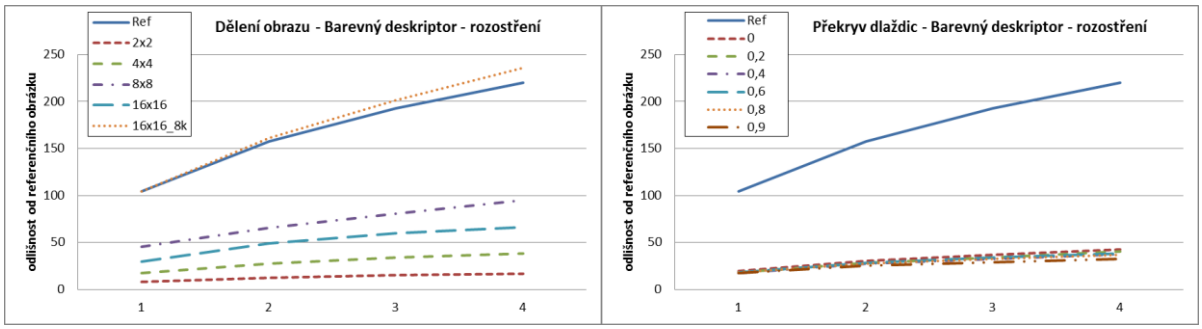


Obrázek 6.9 – Chování deskriptoru při změně posunu obrazu pro různé dělení obrazu (vlevo) a pro různý překryv dlaždic (vpravo)

Test rozostření obrazu

Test změny rozostření obrazu ukazuje, že hrubší dělení obrazu (méně dlaždic) je více tolerantní k rozostření obrazu. Opět je ale vhodné, aby byl deskriptor k určitému rozostření tolerantní, ale dostatečně diskriminující k většímu rozostření. Tento kompromis, jak se zdá, poskytuje dělení 4x4, nebo 8x8.

Test překryvu dlaždic ukazuje, že při změně rozostření obrazu, nemá různý překryv dlaždic na výsledek téměř žádný vliv.

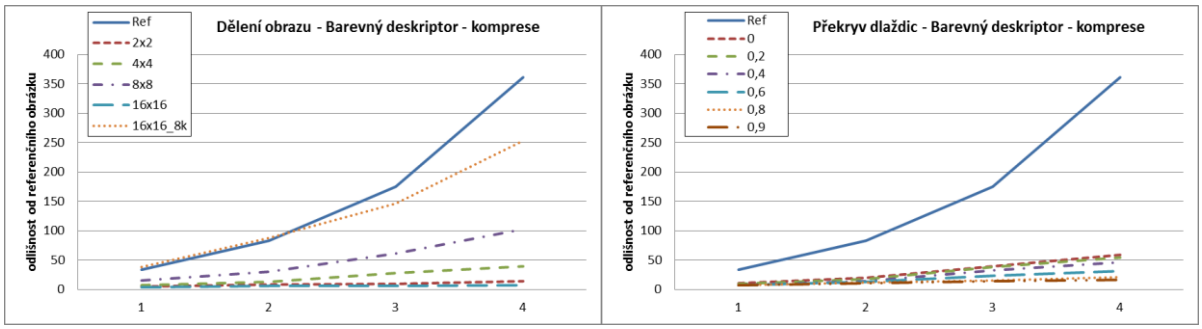


Obrázek 6.10 – Chování deskriptoru při změně rozostření obrazu pro různé dělení obrazu (vlevo) a pro různý překryv dlaždic (vpravo)

Test JPEG komprese obrazu

Je vhodné, aby deskriptor byl ke vlivu komprese obrazu co nejvíce tolerantní. Na druhou stranu je třeba si uvědomit, že obrázek s opravdu výraznou kompresí již lze považovat za rozdílný až odlišný. Z tohoto pohledu vykazuje nejlepší vlastnosti dělení 4x4. Dělení 2x2 a 16x16 jsou příliš tolerantní, na druhou stranu 8x8, či dokonce 16x16_8k jsou již moc „náchylné“ ke vlivu komprese.

Různý překryv dlaždic nemá příliš velký vliv, ale přesto je možné pozorovat, že vhodný kompromis mezi tolerancí a rozlišovací schopností poskytuje překryv 0,4 a 0,6.

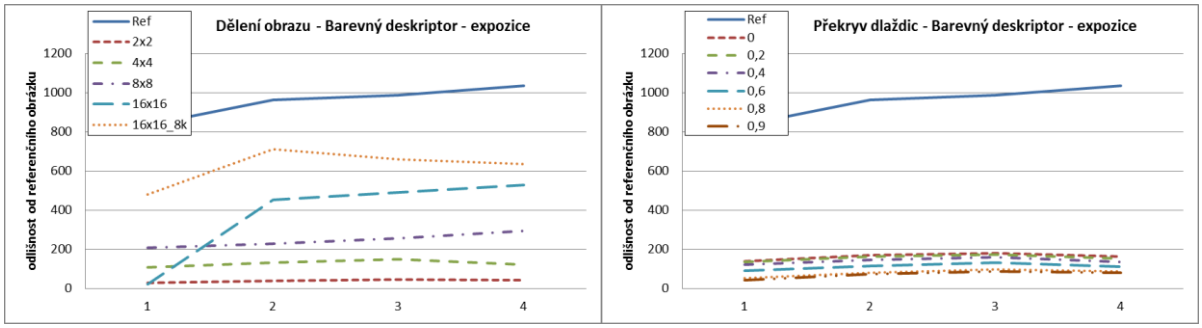


Obrázek 6.11 – Chování deskriptoru při různé úrovni komprese obrazu pro různé dělení obrazu (vlevo) a pro různý překryv dlaždic (vpravo)

Test expozice obrazu

Jelikož pro jemnější dělení obrazu (16x16) vykazuje deskriptor poměrně nekonzistentní výsledky, a pro dělení 2x2 je opět velice tolerantní, vychází jako nejvhodnější hodnoty dělení 4x4, nebo 8x8.

Různý překryv nemá při změně expozice v podstatě žádný vliv (konstantní hodnoty odlišnosti při změně expozice).

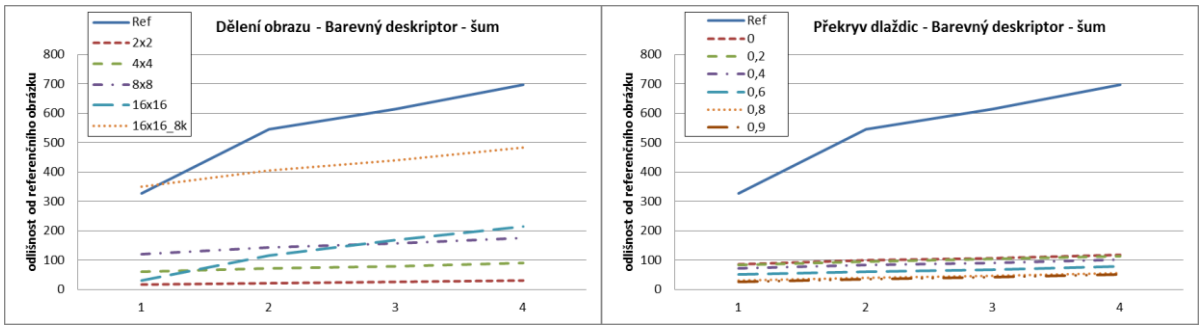


Obrázek 6.12 – Chování deskriptoru při změně expozice obrazu pro různé dělení obrazu (vlevo) a pro různý překryv dlaždic (vpravo)

Test šumu v obraze

Tento test opět ukazuje, že deskriptor je při dělení obrazu na 16x16 velmi náchylný k šumu v obraze a naopak při dělení 2x2 je příliš tolerantní. Proto jako vhodný kompromis vyplývá dělení 4x4 a 8x8.

Různý překryv dlaždic opět nemá vliv na chování deskriptoru při změně úrovně šumu v obraze.



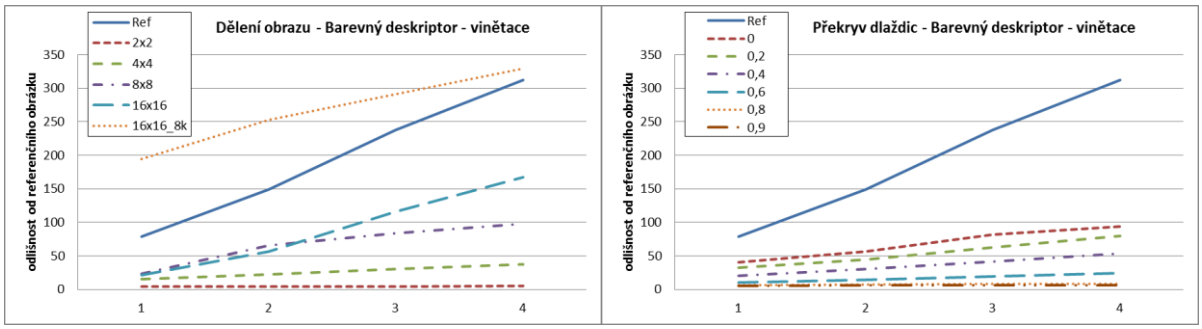
Obrázek 6.13 – Chování deskriptoru při změně množství šumu v obraze pro různé dělení obrazu (vlevo) a pro různý překryv dlaždic (vpravo)

Test vinětace v obraze

Z výsledků testů vyplývá, že čím hrubší dělení obrazu tím má vinětace menší vliv na výsledný deskriptor. To vychází z jeho konstrukce. Jak bylo popsáno v návrhu, jsou hodnoty bodů v dlaždici váhovány Gaussovou funkcí. To znamená, že body na okraji dlaždice mají malý vliv na výsledný deskriptor. Z toho lze usoudit, že pokud by teoreticky byla použita pouze jedna dlaždice pokrývající celý obrázek, byl by vliv vinětace, která se rovněž projevuje na okrajích obrazu, téměř zcela vyrušen váhováním Gaussovou funkcí.

Vinětaci ale není vhodné zcela ignorovat. Proto vychází jako vhodná dělení na 4x4, či 8x8 dlaždic, která jsou dostatečně tolerantní k malé hodnotě vinětace, ale zároveň jsou schopná rozpoznat její vyšší hodnoty.

Podobně je to při různých hodnotách překryvu. Teoretický překryv 1,0 by znamenal, že všechny dlaždice jsou totožné a pokrývají celý obraz, což je v podstatě totožná situace, jako kdyby obraz nebyl dělen (1 dlaždice). Proto čím větší je překryv dlaždic, tím tolerantnější je deskriptor ke vlivu vinětace. Jako vhodný kompromis zde tedy vychází překryv dlaždic 0,4.



Obrázek 6.14 – Chování deskriptoru při různé úrovni vinětace v obraze pro různé dělení obrazu (vlevo) a pro různý překryv dlaždic (vpravo)

Vyhodnocení

Na základě výsledků testování navrhuji optimální parametry deskriptoru jako:

- **dělení obrazu na 4x4 dlaždic**
- **překryv dlaždic 0,5** (50 %)

Takovýto deskriptor je dostatečně tolerantní k menším změnám obrazu, ale zároveň je schopný rozlišit obraz, který by již neměl být považován za „velmi podobný“, ať už je to způsobeno vyšším zkreslením, nebo prostou odlišností obrazu.

Stejným způsobem jsem provedl testy dalších dvou deskriptorů (Gradient, Gradient+RG). Výsledky těchto testů viz Příloha 2. Na jejich základě lze usoudit, že navržené „optimální“ parametry jsou vhodné i pro tyto dva deskriptory. Deskriptory sice dle předpokladu reagují na některé situace rozdílně, ale trend jejich chování ukazuje, že navržené parametry dělení obrazu a překryvu dlaždic jsou pro ně vhodné. Rozborem jednotlivých případů se zde nebudu podrobně zabývat. Předpoklady a způsob vyhodnocení, ale zůstává stejný, jako u analýzy Barevného deskriptoru.

Následující tabulka ukazuje rozložení počtu binů histogramů (1 bin = 1 Byte) pro jednotlivé deskriptory při navržených parametrech.

Deskriptor	**Barevný**			**Gradient**	**Gradient+RG**		
	I	**O1**	**O2**	**Gr**	**Gr**	**R**	**G**
Počet binů histogramu	32	16	16	64	32	16	16

Tabulka 6.1 – Přehled rozložení dat v deskriptorech

6.1.3 Datová sada pro testování vlivu změny obrazu

Pro účely testování jsem pořídil 9 videí tak, aby každé z nich zastupovalo určitý typ videí, jako je sportovní přenos, zpravodajství, kreslený film, apod. Každé z těchto videí je něčím specifické. Některá obsahují rychlý střih, nebo naopak statické scény, či scény s plynulou změnou (např. posun kamery), u některých převažuje určitá barva, některá videa obsahují velice různorodé, či pestré záběry a jiná naopak obsahují záběry s velkými souvislými plochami. Specifika jednotlivých videí shrnuje následující seznam. Každé video má rozlišení 640x360 pixelů a snímkovou frekvenci 25 fps.

- **vid01** – „Climbing a table-top mountain – Expedition Guyana – BBC“
    - Záznam horolezeckého výstupu
    - Záběry na krajinu, skály a jednoho horolezce; Převážně pomalé záběry, místy statické scény; Střih na velmi odlišné scény
    - 5977 snímků
- **vid02** – „Annapurna Base Camp Trekking“
    - Záznam expedice
    - Záběry na skupiny lidí, vesnice, krajinu, řeky a zasněžené hory; Občas vložené statické obrázky
    - 30618 snímků
- **vid03** – „Chile Extreme Whitewater Kayaking“
    - Krátký film o kajakářích na divoké vodě
    - Záběry na kajakáře; Převažuje zpěněná řeka a vodopády; Poměrně rychlý střih na velmi podobné scény; Občas vložené záběry krajiny
    - 8661 snímků
- **vid04** – „Wild Chronicles: Madagascar Poison Frogs“
    - Dokument National Geographic o jedovatých žábách
    - Záběry deštného pralesa a živočichy žijící v něm; Střih ze záběrů na živočichy na scény s výzkumníky; Detailní záběry živočichů a laboratorního vybavení
    - 9111 snímků
- **vid05** – „CNN BREAKING NEWS – NASA SOLAR STORM WARNING“
    - Krátký zpravodajský vstup CNN o sluneční bouři
    - Převážně záběry na moderátory doplněné ilustrační grafikou; Relativně statické scény s pomalým a plynulým pohybem kamery
    - 3048 snímků
- **vid06** – „24H Nürburgring 2011 START“
    - Záznam automobilového závodu
    - Převážně záběry velkého množství závodních automobilů; Rychlý pohyb kamery, místy rozklepaný; Jedná se o záběry z jedné části trati, tudíž se opakují velmi podobné záběry
    - 3264 snímků

- **vid07** – „CORAL REEF“
    - Dokument o korálových útesech
    - Převážně podmořské záběry korálových útesů a živočichů kteří je obývají; Plynulé záběry; Střih na relativně podobné scény; Místy velké stejnobarevné plochy; Titulky na začátku a konci
    - 8638 snímků
- **vid08** – „The Hobbit Trailer“
    - Trailer na film „Hobbit“
    - Převážně záběry na osoby v různých situacích; Velmi rychlý střih na podobně barevně laděné scény; Vložená informační grafika (titulky na začátku, konci i v průběhu videa)
    - 3780 snímků
- **vid09** – „Donald Duck Put-Put Troubles“
    - Krátký kreslený film
    - Rychle se měnící scéna, ale často jen v části obrazu; Velké souvislé plochy; Celé video obsahuje velmi podobně barevně laděné scény
    - 10758 snímků

Zdroj videí – server YouTube.com

Video	vid01	vid02	vid03	vid04	vid05	vid06	vid07	vid08	vid09
počet snímků	5977	30618	8661	9111	3048	3264	8638	3780	10758

Tabulka 6.2 – Délky jednotlivých videí

Pro účely testování deskriptorů jsem tato videa různě modifikoval. Pro každý typ modifikace videa jsem vytvořil tři sady videí se vzrůstající úrovní konkrétní modifikace. Tyto modifikace jsou „**zvětšení+ořez**“, „**změna kvality**“ a „**změna komprese**“. Pro každou modifikaci bylo video zpracováno (modifikováno) 1x, 2x, nebo 3x. Výsledkem jsou tedy vždy tři úrovně této modifikace. Tím jsem získal celkem 90 různých videí. 27 videí pro každou modifikaci a 9 originálních videí

Pro zpracování jsem použil volně dostupný program VirtualDub.

Zvětšení a ořez videa

Každé video bylo zvětšeno na 110, 120 a 130 % (podle úrovně modifikace) a poté oříznuto na původní velikost. Tím bylo video jakoby přiblíženo se současnou ztrátou informace na okrajích obrazu.

Úroveň	Velikost [%]	Ořez [ŠxV]
1	110	640x360
2	120	640x360
3	130	640x360

Tabulka 6.3 – Přehled hodnot zvětšení a ořezu videa

Obrázek 6.15 – Ilustrace úpravy „zvětšení+ořez“ pro 1. snímek videa „vid01“

Změna „kvality“ videa

Každému videu byl zvýšen jas na 105, 110 a 115 % se současným snížením kontrastu na 85, 70 a 60 % a gaussovským rozostřením s poloměrem 2, 4 a 6 pixelů. Tím je simulováno snížení „kvality“ videa. Hodnoty modifikace pro jednotlivé úrovně ukazuje následující tabulka.

Úroveň	Modifikace		
	Jas [%]	Kontrast [%]	Rozostření [px]
1	105	85	2
2	110	70	4
3	115	60	6

Tabulka 6.4 – Přehled hodnot úprav videa pro změnu „kvality“

Obrázek 6.16 – Ilustrace změny „kvality“ pro 1. snímek videa „vid01“

Komprese videa

Videa jsem komprimoval kodekem DivX s nastavením pro získání co nejmenšího souboru. Pro získání následující úrovně jsem pak se stejným nastavením komprimoval již zpracované video, čímž jsem dosáhl další komprimace a zhoršení kvality výsledného videa. Tabulka 6.5 ukazuje datový tok jednotlivých videí při různých úrovních komprese.

Úroveň	Datový tok [kbps]								
	vid01	**vid02**	**vid03**	**vid04**	**vid05**	**vid06**	**vid07**	**vid08**	**vid09**
originál	794	791	786	788	796	792	790	772	787
1	164	180	389	200	218	463	266	170	325
2	138	151	321	171	187	416	223	148	284
3	125	136	287	155	168	374	201	137	263

Tabulka 6.5 – Přehled hodnot datového toku při různých úrovních komprese videa

Obrázek 6.17 – Ilustrace různých úrovní komprese pro 1. snímek videa „vid01“

6.1.4 Testování vlivu zkreslení obrazu na práci deskriptorů

Pro účely testování používám takzvanou matici podobností (similarity matrix). Tato matice *S* vznikne obecně porovnáním dvou videí *S(V1, V2, a)*, přičemž každý prvek matice S_{xy} je výsledkem porovnání snímku *y* z videa *V1* se snímkem *x* z videa *V2* použitím deskriptoru typu *a*. Každý řádek tedy reprezentuje porovnání jednoho snímku z videa *V1* se všemi snímky z videa *V2*. Porovnávají se deskriptory snímků s použitím L_2 vzdálenosti.

$$S(V1, V2, a)_{xy} = L_2\left[D_a\left(V1_y\right), D_a(V2_x)\right] \tag{19}$$

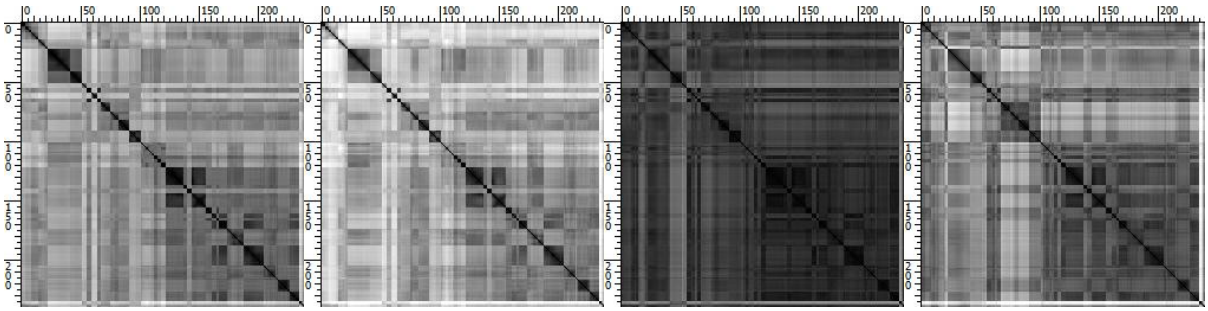
kde $D_a()$ je deskriptor typu *a*, $V1_y$ *y*-tý snímek videa *V1* a $V2_x$ je *x*-tý snímek videa *V2*. Výsledná matice má tedy výšku rovnou počtu snímků videa *V1* a analogicky šířku rovnou počtu snímků videa *V2*.

V první fázi testování jsem vytvořil pro každé video matici podobností tak, že jsem ho porovnal se sebou samým a to pro každý typ deskriptoru – *S(V1, V1, a)*. Vzhledem k velkému množství snímků ve videích jsem porovnával každý 25. snímek (neboli každou vteřinu videa). Na několika vzorcích jsem experimentálně zjistil, že takovýto přístup je sice méně přesný, ale výsledná chyba je < 1 % a je tedy zanedbatelná.

Obrázek 6.18 ukazuje takto vypočítané matice pro stejné video, ale různými deskriptory. Černá barva znamená, že snímky jsou totožné (nulový rozdíl) naopak bílá označuje velice rozdílné snímky. Proto, čím tmavší barva bodu, tím podobnější jsou dva porovnávané snímky videa. Z konstrukce

matice podobností vyplývá, že identické snímky jsou porovnávány na diagonále (první s prvním až poslední s posledním), což také potvrzují výsledné matice. Na každé je jasně patrná černá diagonální linie, která v podstatě označuje průběh videa. Dále je možné pozorovat, že statický (či relativně statický) záběr, kdy jsou navazující snímky velice podobné, se projeví tmavým „čtvercem“ umístěným na diagonále.

Takovéto matice jsem vytvořil pro každé originální video a každý deskriptor. Tím jsem získal 36 matic podobností.

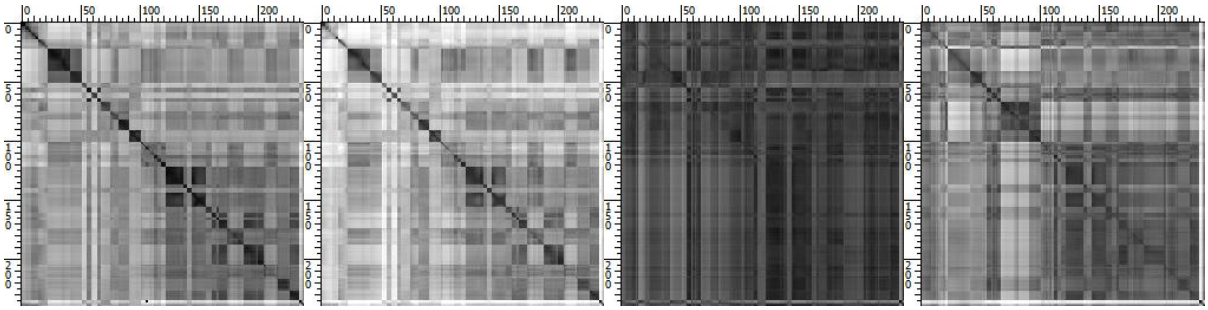


Obrázek 6.18 – Matice podobností pro video „vid01“ vypočítané pro jednotlivé deskriptory. Zleva - Referenční, Barevný, Gradient a Gradient+RG deskriptor.

Ve druhé fázi jsem vypočítal matice podobností originálního videa s modifikovaným videem (každý snímek originálního videa se porovná s každým snímkem modifikovaného videa). Neboli při zachování předchozí konvence – $S(V_{orig}, V_{modifXY}, a)$, kde *X* je typ modifikace a *Y* je úroveň modifikace *[1, 2, 3]* a *a* je použitý typ deskriptoru.

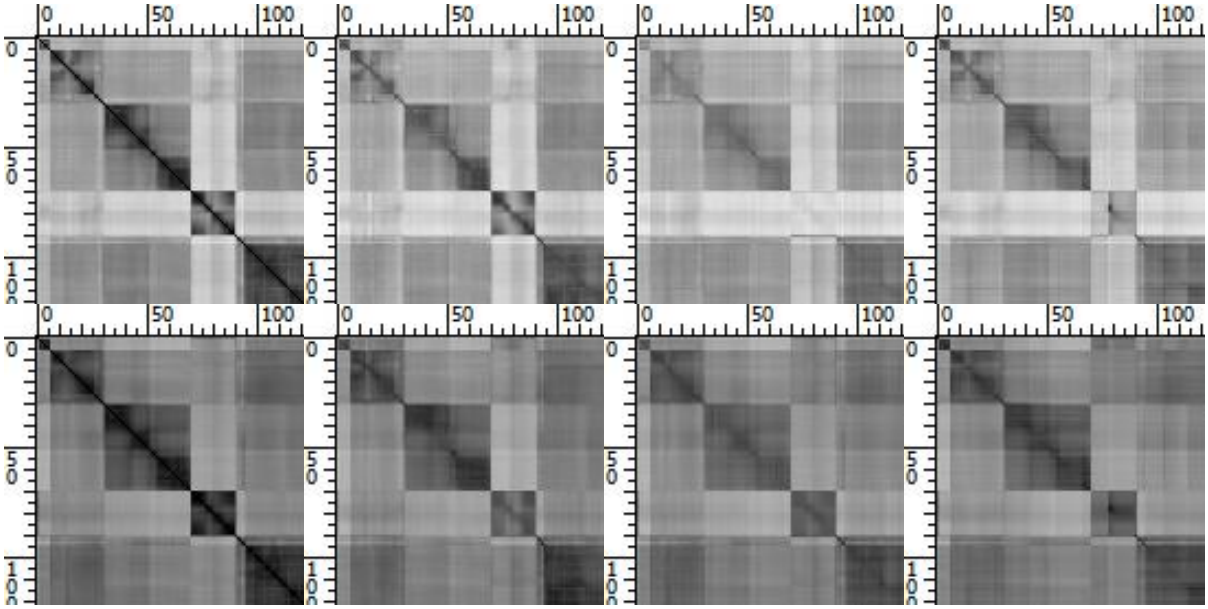
Každé video jsem porovnal se všemi jeho modifikovanými variantami s použitím všech čtyř deskriptorů. Modifikace videí byly popsány v předchozí části práce. Jak je uvedeno, pro každé video jsem vytvořil 9 modifikovaných variant (3 typy modifikace o 3 úrovních). To pro 9 videí dává dohromady 81 matic podobností. Pro 4 deskriptory to tedy znamená celkem 324 matic podobností. Celkem jsem tedy získal i s výsledky z první fáze **360 matic podobností.**

Obrázek 6.19 ukazuje takovéto matice podobností pro video „vid01“ které bylo porovnáno s jeho modifikovanou variantou (komprese úrovně 3). Je zde vidět, že tato modifikace má vliv hlavně na poslední dva deskriptory využívající gradient. To plyne ze skutečnosti, že vysokou kompresí videa utrpěly některé méně významné hrany v obraze. Tím pádem toto zkreslení hodně ovlivnilo gradienty v obraze.



Obrázek 6.19 – Porovnání videa „vid01“ s jeho modifikovanou verzí „komprese – 3“. Zleva – Referenční, Barevný, Gradient a Gradient+RG deskriptor.

Obrázek 6.20 ukazuje matice podobností videa „vid05“ pro všechna zkreslení videa úrovně 3 pro Barevný a Referenční deskriptor. Jak je vidět, je zachována struktura videa (tmavší čtverce kolem diagonály), ale vytrácí se podobnost snímků. V některých maticích dokonce zaniká, či splývá diagonála označující snímky, které mohou být označeny za stejné (kvůli zkreslení stejné nejsou).



Obrázek 6.20 – Matice podobností videa „vid05“ s jeho modifikacemi úrovně 3. Zleva – originál, „zvětšení+ořez“, „kvalita“, „komprese“. Nahoře pro Barevný deskriptor a dole pro Referenční deskriptor.

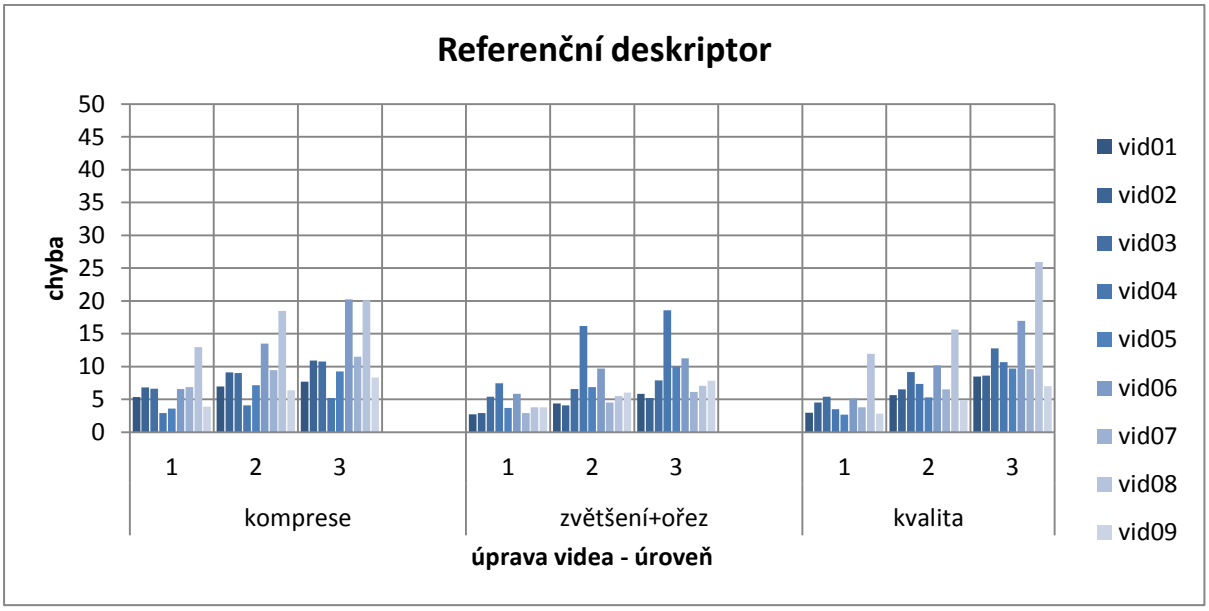
V poslední fázi jsem získané matice porovnal. Samozřejmě jsem porovnával pouze matice pro stejná videa. Výsledkem je tedy vždy rozdílnost matice podobností zkresleného videa od matice podobností originálního videa.

Výsledkem porovnání dvou matic je součet absolutních rozdílů korespondujících prvků matice dělený velikostí matice (kvůli zamezení vlivu délky videa na absolutní hodnotu výsledku).

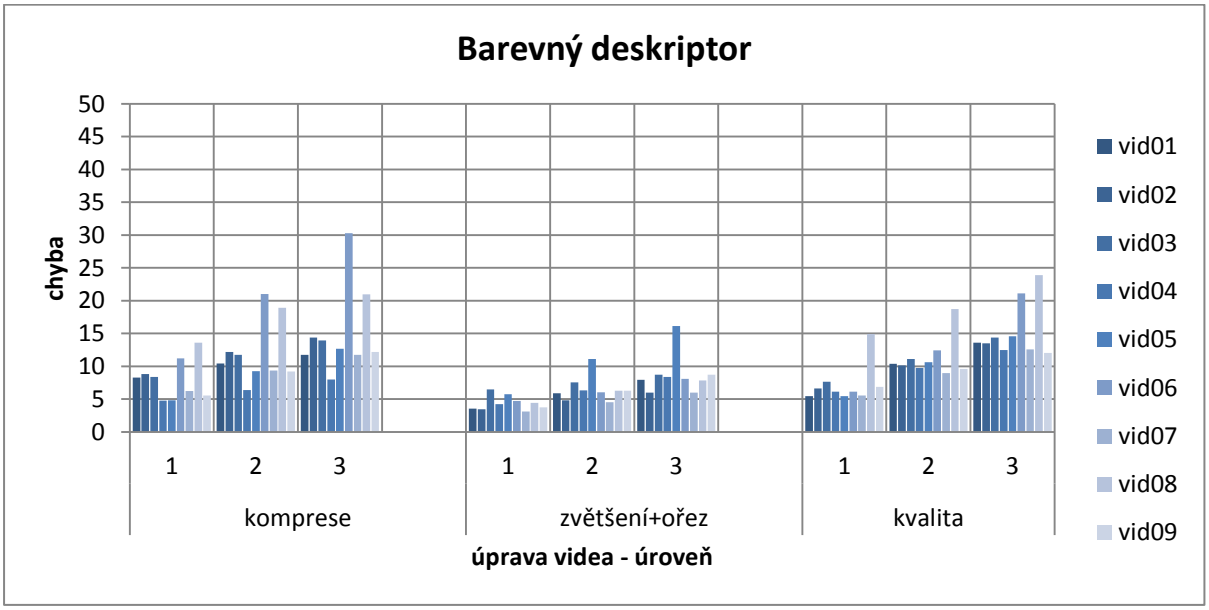
$$R(S1,S2) = \frac{\sum_{y=1}^{Y}\sum_{x=1}^{X}|S1_{xy}-S2_{xy}|}{XY} \tag{20}$$

kde *R* je výsledek porovnání matic podobností (rozdíl, či chyba) a *X*, *Y* jsou rozměry matic. Jelikož se porovnávají vždy matice pro videa o stejné délce, jsou opravdu tyto rozměry pro obě matice stejné.

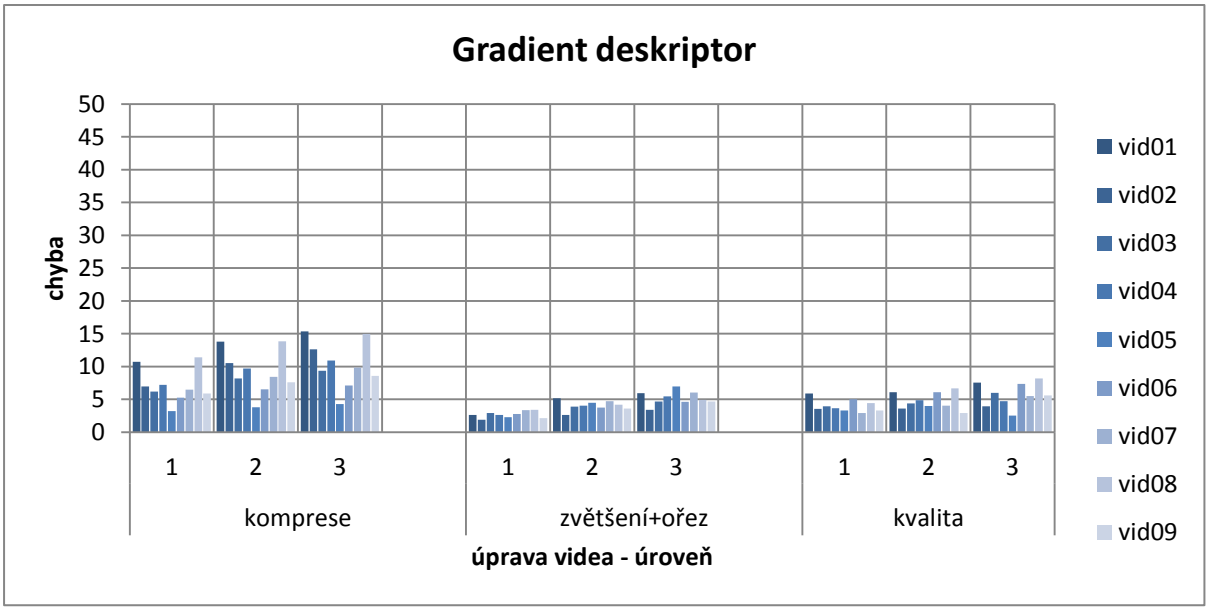
Výsledky porovnání matic podobností jsou zaznamenány v následujících diagramech. Diagramy jsou rozděleny do tří částí, přičemž každá část ukazuje vývoj chyby (výsledku porovnání) pro tři úrovně konkrétní modifikace videí.



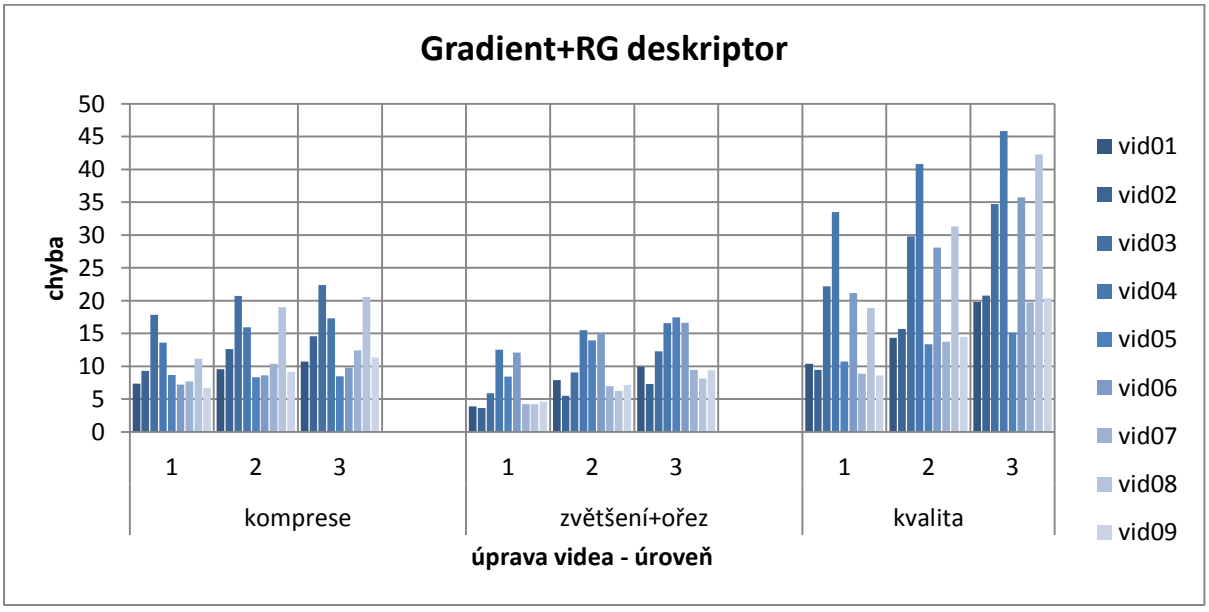
Obrázek 6.21 – Porovnání matic podobností modifikovaných videí s maticemi podobností originálních videí pro Referenční deskriptor



Obrázek 6.22 – Porovnání matic podobností modifikovaných videí s maticemi podobností originálních videí pro Barevný deskriptor



Obrázek 6.23 – Porovnání matic podobností modifikovaných videí s maticemi podobností originálních videí pro Gradient deskriptor



Obrázek 6.24 – Porovnání matic podobností modifikovaných videí s maticemi podobností originálních videí pro Gradient+RG deskriptor

Vyhodnocení

Na výsledcích porovnání je vidět, že Barevný deskriptor podává srovnatelné výsledky s Referenčním deskriptorem. To lze předpokládat jelikož pracuje se stejným barevným modelem. Na základě těchto testů lze tedy říci, že Barevný deskriptor má podobné rozlišovací schopnosti jako Referenční deskriptor.

Dále je vidět, že deskriptor využívající gradienty je tolerantní ke změnám „kvality“ videa, které ale zachovává strukturu obrazu a současně i ke změně „zvětšení+ořez“. Naopak „komprese“ obrazu se již projeví vyšší hodnotou chyby. Dá se říci, že je tedy vhodný pro případy, kde lze očekávat různé hodnoty jasu, kontrastu, či rozdíly v barevnosti, ale kde je zachována strukturální informace.

Pro deskriptor Gradient+RG je změna „kvality“ naopak velmi zásadní, jelikož pracuje s normovanou barevnou informací a proto se projeví velkou hodnotou chyby. Lze tedy říci, že je vhodný pro případy, kde je třeba rozpoznat podobnost na základě barevnosti i struktury, ale kde není třeba přílišná míra tolerance ke zkreslení obrazu.

6.2 Testování metody vyhledávání

Pro účely testování vyhledávání jsem vytvořil další datovou sadu, kterou popisuji v první části této kapitoly. Druhá část popisuje samotné testování vyhledávání a vyhodnocení výsledků.

6.2.1 Datová sada pro testování vyhledávání

Pro účely testování vyhledávání jsem z původních videí použitých v předchozí části vytvořil takzvaná konkatenovaná (zřetězená) videa. Jedná se vždy o spojení dvou videí způsobem *A+B+A*. Operátorem *+* zde označuji konkatenaci. Například pro *A = vid01* a *B = vid02* je výsledné video *vid_121* spojením *vid01 + vid02 + vid01*. Takže video *A* se ve výsledném videu vyskytuje 2krát. Tato skutečnost je anotací pro konkatenovaná videa a mohu je tedy použít pro testování vyhledávání.

Následující tabulka ukazuje počet snímků výsledných videí a údaje, kolik procent výsledného videa zabírá zdrojové video *A* (je obsaženo 2krát) a zdrojové video *B*.

Video	vid_121	vid_232	vid_343	vid_454	vid_565	vid_676	vid_787	vid_898	vid_919
snímků	42572	69897	26433	21270	9360	15166	21056	18318	27493
video A [%]	28,1	87,6	65,5	85,7	65,1	43,0	82,0	41,3	78,3
video B [%]	71,9	12,4	34,5	14,3	34,9	57,0	18,0	58,7	21,7

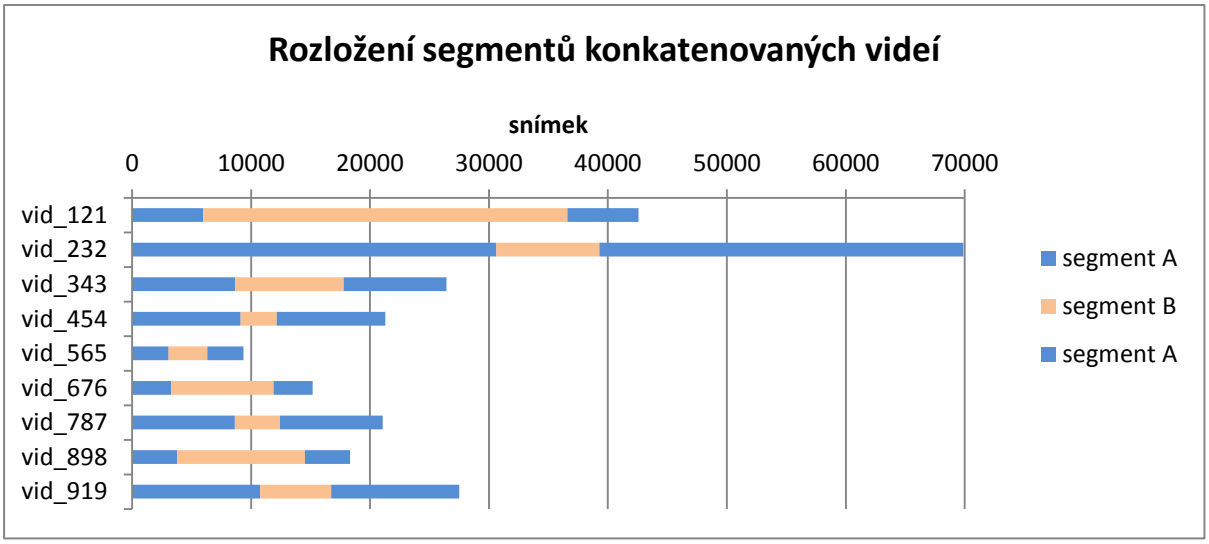
Tabulka 6.6 – Přehled počtu snímků konkatenovaných videí a procentní zastoupení jednotlivých segmentů ve videu (*video A* = oba segmenty videa *A*)

Anotace konkatenovaných videí ukazuje následující tabulka (Tabulka 6.7), kde jsou zaznamenány počáteční a koncové snímky jednotlivých videí.

Snímek	vid_121	vid_232	vid_343	vid_454	vid_565	vid_676	vid_787	vid_898	vid_919
start A1	0	0	0	0	0	0	0	0	0
end A1	5976	30617	8660	9110	3047	3263	8637	3779	10757
start B	5977	30618	8661	9111	3048	3264	8638	3780	10758
end B	36594	39278	17771	12158	6311	11901	12417	14537	16734
start A2	36595	39279	17772	12159	6312	11902	12418	14538	16735
end A2	42571	69896	26432	21269	9359	15165	21055	18317	27492

Tabulka 6.7 – Přehled začátků a konců jednotlivých segmentů v konkatenovaných videích

Grafické znázornění rozložení segmentů v jednotlivých konkatenovaných videích ukazuje následující diagram (viz Obrázek 6.25). Je zde vidět celková délka videí a délka jednotlivých segmentů.



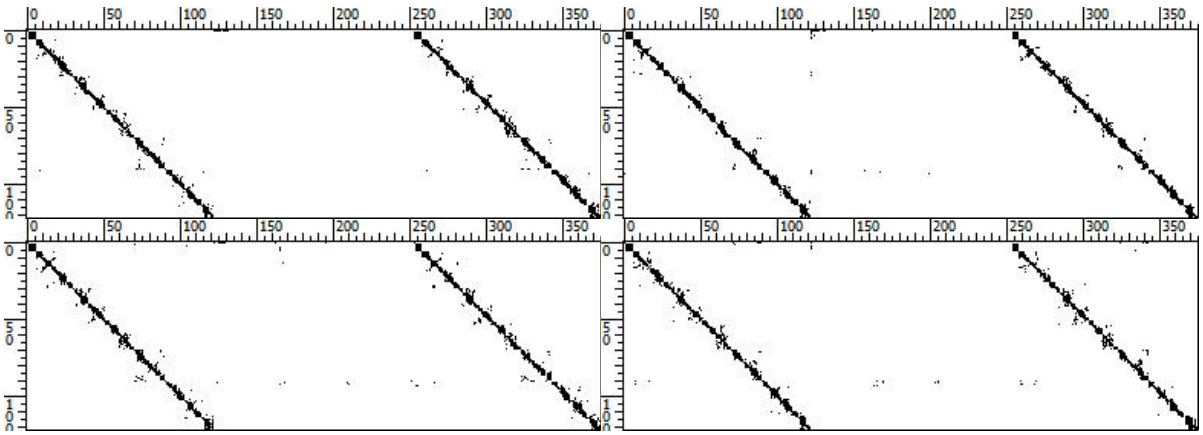
Obrázek 6.25 – Ilustrace rozložení segmentů v konkatenovaných videích a porovnání jejich celkové délky

6.2.2 Testování vyhledávání

K testování vyhledávání jsem opět využil matici podobností definovanou v části 6.1.4. V tomto případě, ale záznam v matici neukazuje podobnost dvou snímků, ale slouží pro zaznamenání nalezeného „podobného“ snímku.

Pro každý snímek z videa *vid0A* je s použitím LSH vyhledáno 10 nejpodobnějších snímků v konkatenovaném videu *vid_ABA*. Stejně jako v předchozí kapitole byl použit každý 25. snímek videa.

Obrázek 6.26 ukazuje takto vypočítané matice pro srovnání videa „vid05“ s videem „vid_565“. Na výsledku jsou jasně patrné dva rozpoznané segmenty videa *A* (vid01), mezi které je vloženo video *B* (vid02), které se na výsledku projevilo jako prázdná oblast (cca 120 – 250). Zároveň je patrné, že pro tato videa jsou výsledky různých deskriptorů velice podobné.



Obrázek 6.26 – Matice podobností videa „vid05“ (osa y) s videem „vid_565“ (osa x) získané pomocí LSH, pro všechny deskriptory. Nahoře: Referenční, Barevný. Dole: Gradient, Gradient+RG.

Tyto matice podobností jsem vytvořil pro každé konkatenované video (vid_121 – vid_919), pro každý deskriptor. Tím jsem získal celkem 36 matic podobností.

Je třeba poznamenat, že jsem srovnával videa „vid0A“ s videi „vid_ABA“, neboli vyhledával jsem dva segmenty v konkatenovaném videu. Bylo by možné také porovnat videa „vid0B“ s videi „vid_ABA“, neboli vyhledávat jeden segment uprostřed konkatenovaného videa. Toto porovnání ale považuji za komplement předchozího porovnání (tudíž za nadbytečné).

Na získané matice jsem poté aplikoval detekci souvislých segmentů, kterou jsem implementoval do aplikace, jejíž výstupem je textový popis nalezených segmentů ve videu. Výstupem je ID nalezeného segmentu, začátek a konec segmentu ve videu A a začátek a konec segmentu ve videu B a „score“, které udává vypočítanou jistotu správnosti nalezení segmentu. Pozice ve videu jsou vyjádřeny ve snímcích (příklad viz Tabulka 6.8).

ID	start_A	stop_A	start_B	stop_B	score
Segment0	0	5975	0	6000	0,917012
Segment1	0	5975	36600	42550	0,945833

Tabulka 6.8 – Příklad nalezených segmentů videa (A = vid01, B = vid_121) pro deskriptor Gradient+RG

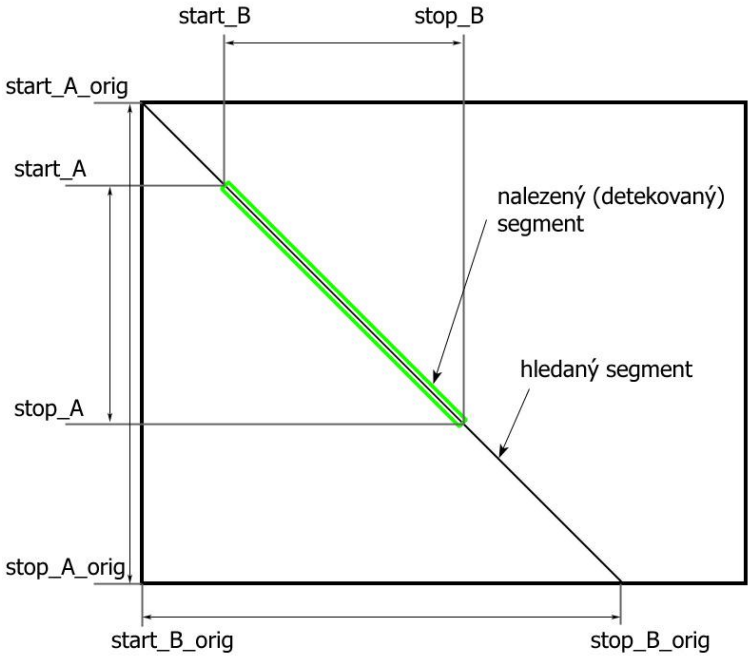
Takto jsem získal umístění nalezených segmentů pro všechna testovaná videa. Z výsledků jsem vždy vypočítal absolutní hodnotu chyby nalezeného segmentu, neboli o kolik se liší hodnoty nalezeného segmentu od anotovaných hodnot skutečné pozice segmentu (hodnoty „start A1“, „end A1“, „start A2“, „end A2“ viz Tabulka 6.7.)

Výpočet chyby jednoho segmentu (viz Obrázek 6.27) je určen na základě následujícího vzorce:

$$CHYBA = (|start\_A\_orig - start\_A| + |stop\_A\_orig - stop\_A| + |start\_B\_orig - start\_B| + |stop\_B\_orig - stop\_B|) \cdot score \quad (21)$$

Je zřejmé, že *start_A_orig* je v tomto případě vždy roven *0*, neboť se jedná o začátek videa *A* a *stop_A_orig* je konec (neboli celkový počet snímků) videa *A*. Hodnoty *start_B_orig* a *stop_B_orig* jsou zmiňované hodnoty z tabulky anotací (viz Tabulka 6.7).

Hodnotu chyby jsem vypočítal vždy pro každý segment videa zvlášť.



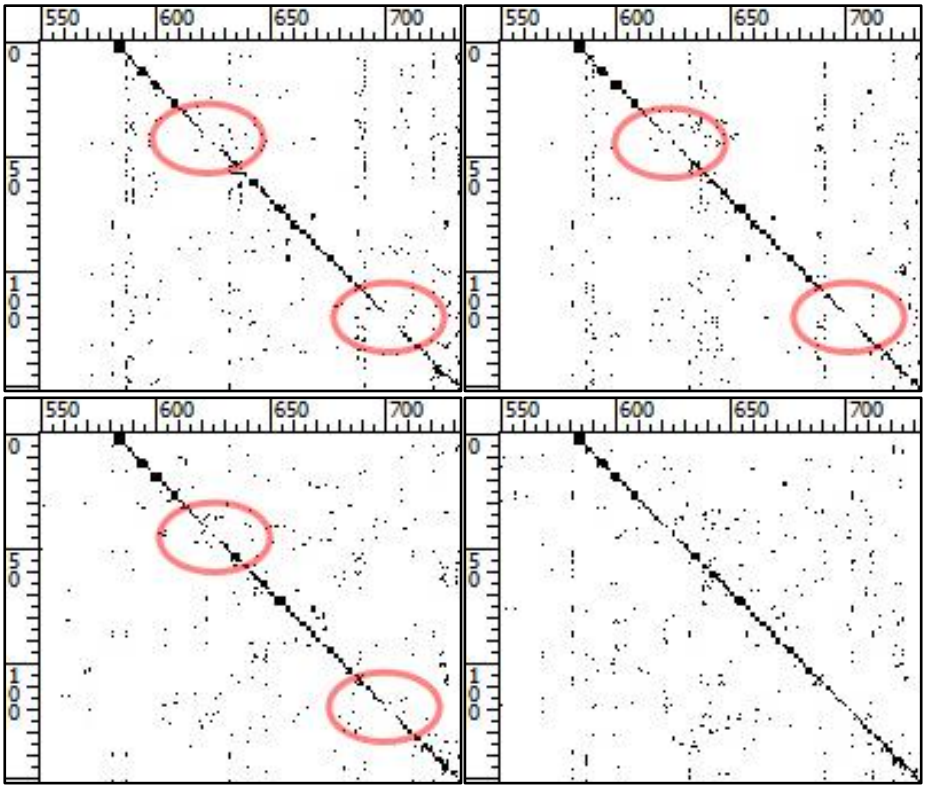
Obrázek 6.27 – Ilustrace chyby detekovaného segmentu videa

Vypočítané hodnoty chyby jsem zanesl do diagramů (viz Obrázek 6.29). Tyto diagramy ukazují velikost chyby detekce pro každý segment videa zvlášť. První segment je zakreslen modrou a druhý červenou barvou. Jednotlivá vyhledání jsou v diagramech označena pro přehlednost (1..9), přičemž „1“ označuje vyhledání videa „vid01“ ve videu „vid_121“ a obdobně pro další.

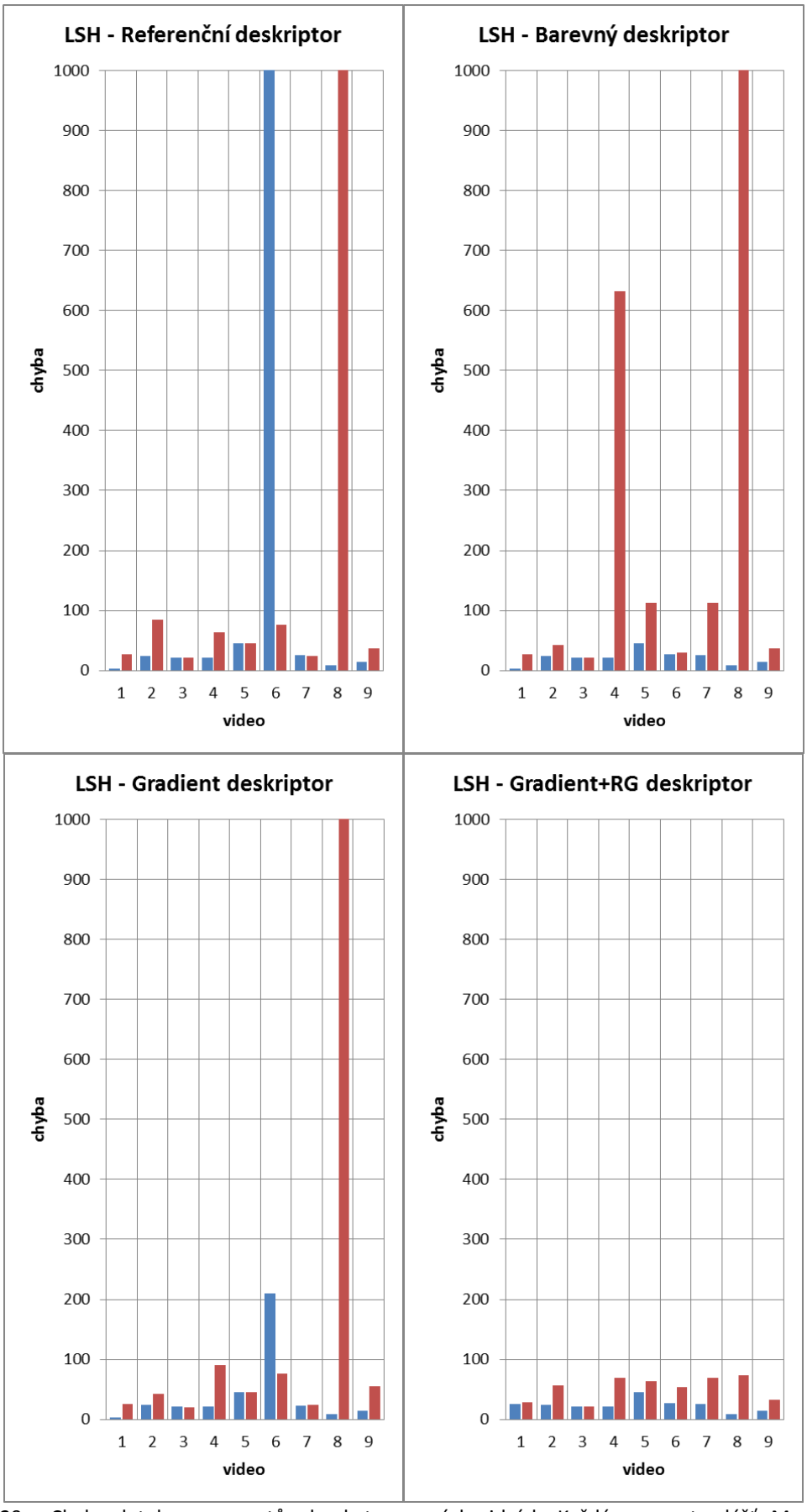
Osa „chyba“ je v diagramech záměrně omezena maximální hodnotou 1000, jelikož větší chybu zde považuji za chybu absolutní a proto není důležitá její přesná velikost. Naopak zajímavé jsou menší hodnoty chyby. Z videí byl uvažován každý 25. snímek, proto je také elementární jednotka chyby *25*. Proto může být chyba menší než 50 považována za přesné rozpoznání segmentu.

Z diagramů je vidět, že největší problémy měly deskriptory s rozpoznáním segmentů videa „vid08“, kde oba segmenty nalezl úspěšně pouze jeden deskriptor (Gradient+RG). Ostatní deskriptory nalezly pouze první segment. Obrázek 6.28 ukazuje matice podobností pro tyto vyhledání. Je zde vidět, že diagonální linie je u prvních tří deskriptorů na dvou místech přerušena, což vedlo k chybě detekce segmentu. Pouze pro poslední deskriptor (Gradient+RG) je linie souvislá.

Nejlepší výsledky v této situaci tedy jasně podává deskriptor „Gradient+RG“. V případě ostatních deskriptorů se ale jedná o ojedinělé problémy. Každý deskriptor dokázal přesně určit vždy jeden segment a většinou i segment druhý.



Obrázek 6.28 – Vyhledání videa „vid08“ ve videu „vid_898“ pro všechny deskriptory. Zobrazen pouze výřez na 2. segment. Nahoře: Referenční, Barevný. Dole: Gradient, Gradient+RG.

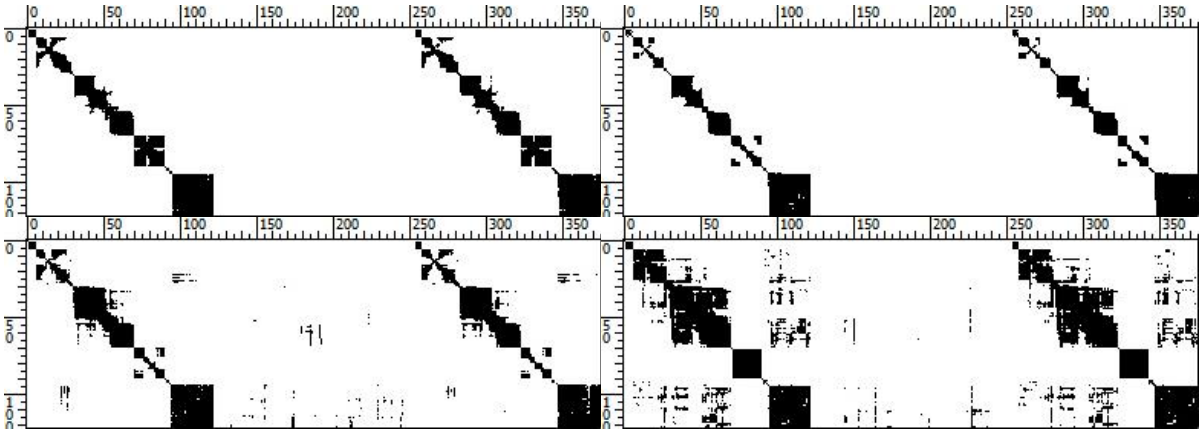


Obrázek 6.29 – Chyba detekce segmentů v konkatenovaných videích. Každý segment zvlášť. Modrá – první segment, červená – druhý segment.

Bez použití LSH

Pro srovnání jsem stejné testy provedl i bez použití LSH a to tak, že jsem podobně jako v kapitole 6.1.4 porovnal všechny snímky z videa A se všemi snímky konkatenovaného videa ale na výslednou matici podobností jsem následně aplikoval operaci prahování, přičemž hodnota prahu byla automaticky určena na základě hodnot v matici.

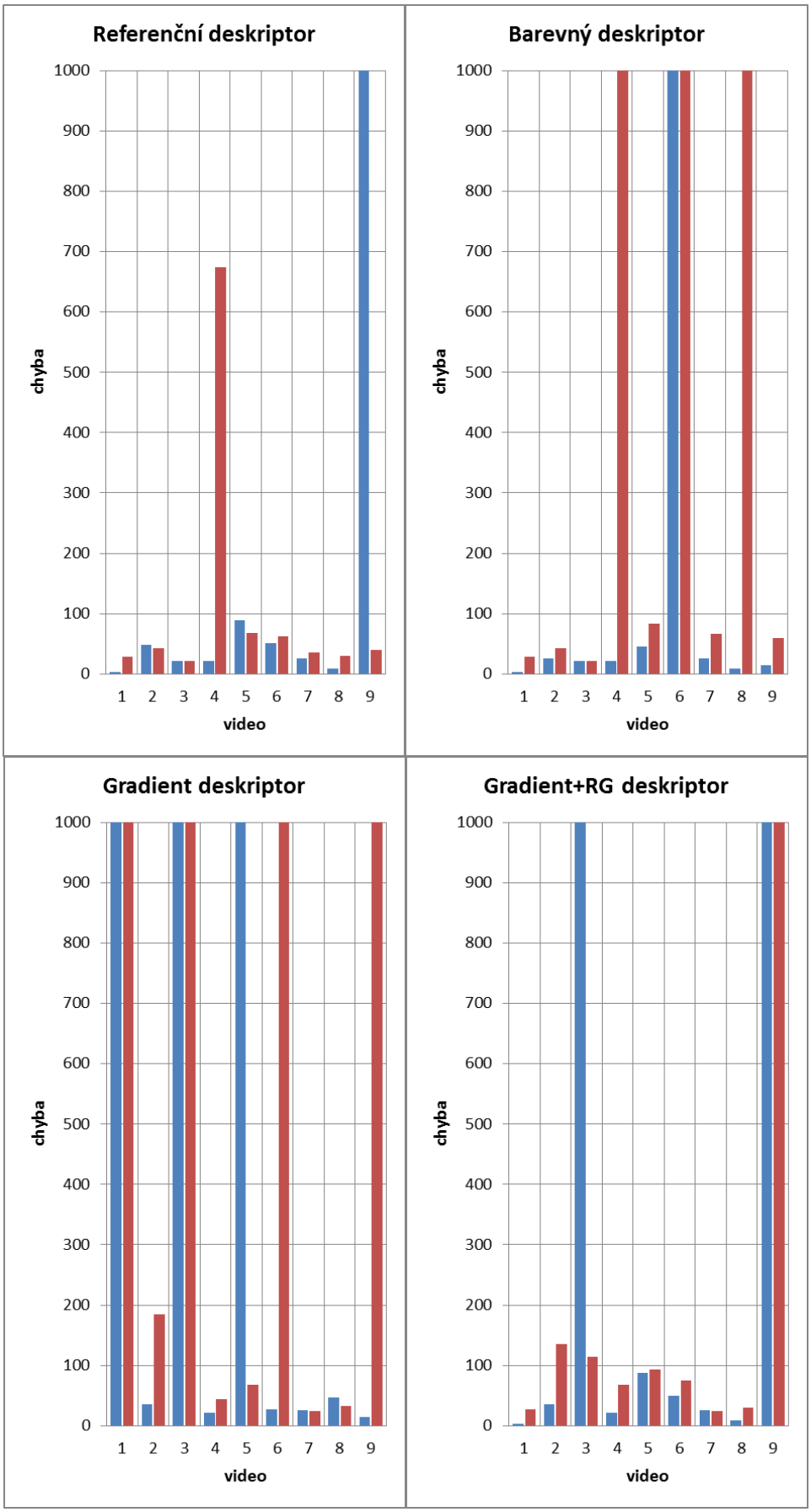
Obrázek 6.30 ukazuje příklad takto získaných matic pro stejný případ jako s použitím LSH (viz Obrázek 6.26), z kterého je jasně vidět vyšší přesnost detekce při použití LSH.



Obrázek 6.30 – Matice podobností videa „vid05“ (osa y) s videem „vid_565“ (osa x) získané pomocí porovnání všech snímků a následného prahování, pro všechny deskriptory. Nahoře: Referenční, Barevný. Dole: Gradient, Gradient+RG.

Následující diagramy (viz Obrázek 6.31) ukazují hodnoty chyby pro všechny typy deskriptorů při vyhledávání podobných segmentů metodou hrubé síly, tedy bez použití LSH.

Je vidět, že Referenční deskriptor podává podobné výsledky jako s použitím LSH (i když selhává u jiných videí, než s použitím LSH), ale pro ostatní tři deskriptory jsou výsledky výrazně horší.

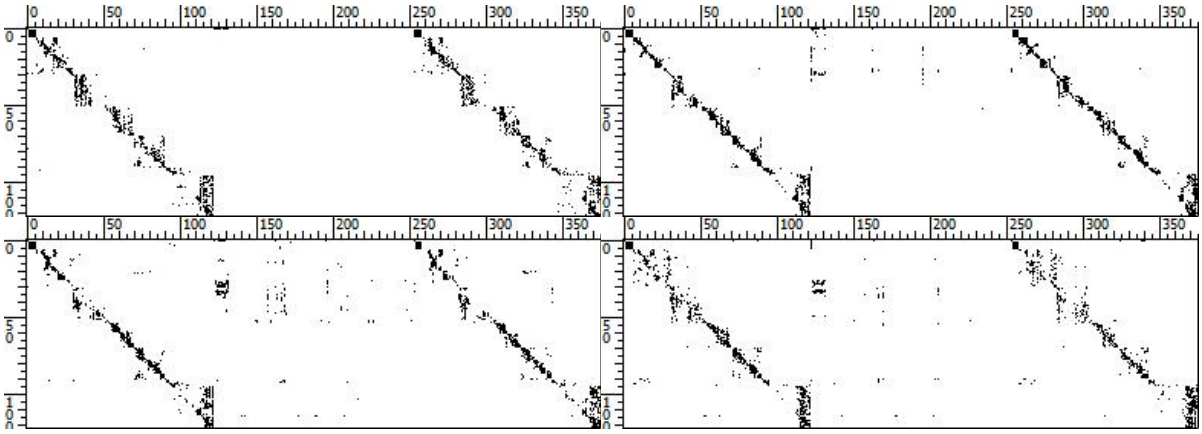


Obrázek 6.31 – Chyba detekce segmentů v konkatenovaných videích bez použití LSH. Modrá – první segment, červená – druhý segment.

Vyhledání modifikovaných videí

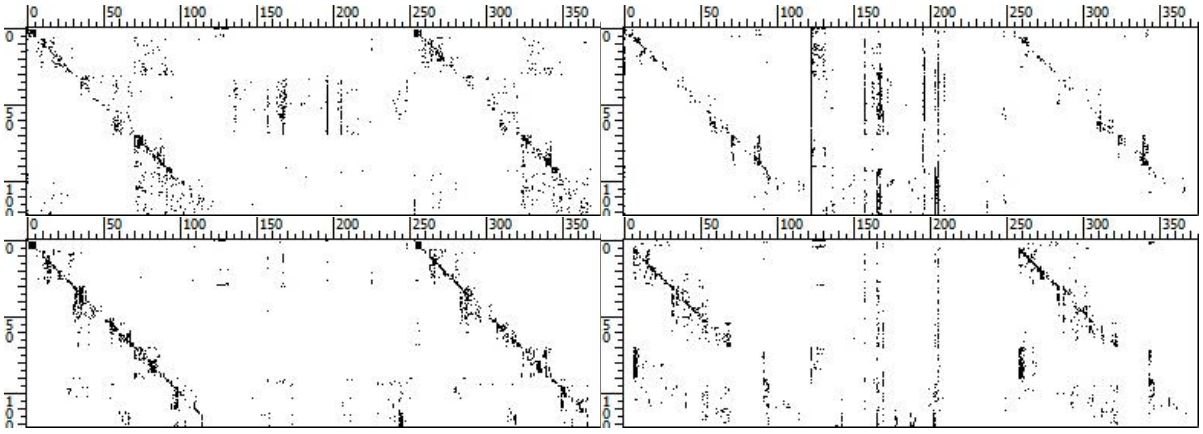
Nakonec jsem provedl stejné testy pro modifikovaná videa použitá v kapitole 6.1.3. Neboli testoval jsem vyhledání segmentů v konkatenovaných videích na základě modifikovaného originálního videa. Do srovnání jsem zahrnul pouze videa „vid01“, „vid05“ a „vid09“ s každým typem modifikace úrovně 3 (celkem 9 videí). To vše opět pro všechny čtyři deskriptory. Dohromady tedy 36 vyhledání.

Obrázek 6.32 ukazuje příklad vypočítaných matic podobností pro vyhledání segmentů videa „vid05“ s modifikací „zvětšení+ořez“ úrovně 3 ve videu „vid_565“. Obrázek ukazuje matice vypočítané pro všechny čtyři deskriptory. Je jasně vidět zkreslení oproti vyhledání nemodifikovaného videa (viz Obrázek 6.26), ale stále je patrná struktura segmentů.



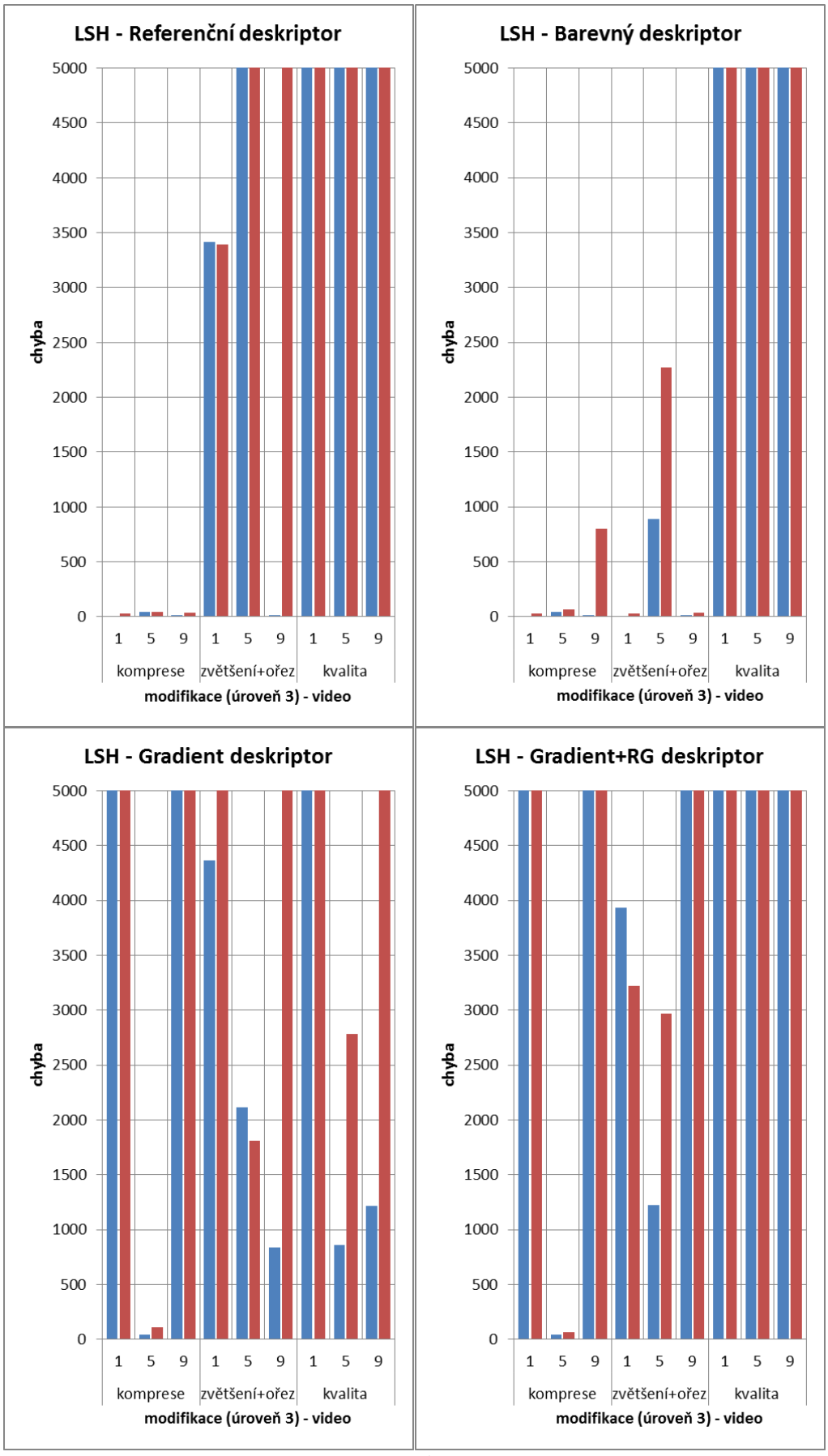
Obrázek 6.32 – Matice podobností videa „vid05“ modifikovaného „zvětšení+ořez“ úrovně 3 (osa y) s videem „vid_565“ (osa x) získané pomocí LSH, pro všechny deskriptory. Nahoře: Referenční, Barevný. Dole: Gradient, Gradient+RG.

Oproti tomu, při vyhledání segmentů videa s modifikací „kvalita“ úrovně 3 je vidět (viz Obrázek 6.33), že tato modifikace má již na vyhledání opravdu výrazný vliv. Nejlépe v tomto případě pracuje deskriptor využívající pouze gradienty (Gradient deskriptor), což není překvapivé vzhledem k faktu, že ostatní deskriptory jsou velmi závislé na barevné informaci v obraze, která je změnou jasu a kontrastu významně ovlivněna. Samozřejmě kvůli současnému rozostření obrazu utrpěly touto změnou i gradienty obrazu, ale toto zkreslení zde není tak výrazné, jako zkreslení barevné a jasové informace.



Obrázek 6.33 – Matice podobností videa „vid05“ modifikovaného „kvalita“ úrovně 3 (osa y) s videem „vid_565“ (osa x) získané pomocí LSH, pro všechny deskriptory. Nahoře: Referenční, Barevný. Dole: Gradient, Gradient+RG.

Výsledky testů jsou vidět na následujících diagramech (viz Obrázek 6.34). Diagramy opět ukazují velikost chyby při vyhledání jednotlivých segmentů modifikovaných videí v konkatenovaných videích. Z výsledů testů lze usoudit, že při celkovém porovnání poskytuje nejlepší výsledky Barevný deskriptor. Je ale třeba poznamenat, že Gradient deskriptor poskytuje nejlepší výsledky při vyhledání segmentů se zhoršenou „kvalitou“. Referenční deskriptor zde spolu s Barevným deskriptorem vynikají při rozpoznání segmentů při vysoké kompresi obrazu.



Obrázek 6.34 – Chyba detekce modifikovaných segmentů v konkatenovaných videích s použitím LSH. Modrá – první segment, červená – druhý segment.

7 Závěr

Cílem této práce bylo navrhnout metodu popisu obrazu s využitím barevných histogramů a její použití pro vyhledávání podobných obrázků. Při návrhu jsem vycházel z teoretických poznatků popsaných v první části práce. Navrhl jsem globální deskriptor, který využívá barevné histogramy v oponentním barevném modelu a jeho dvě modifikace, které navíc využívají histogramy gradientů.

V jazyce C++, s využitím OpenCV, jsem implementoval vlastní knihovnu funkcí pro výpočet navržených deskriptorů a deskriptoru použitého v metodě o kterou jsem se při návrhu opíral. Tento deskriptor jsem použil jako referenční při vyhodnocování výsledků testování.

Dále jsem implementoval aplikaci pro účely testování deskriptorů a na základě výsledků testů jsem navrhl jejich ideální parametry. Deskriptory s těmito parametry jsem poté testoval pro zhodnocení jejich použitelnosti a vhodnosti pro různé případy použití.

V poslední fázi jsem otestoval metodu, která používá navržené deskriptory pro popis obrazu a indexační metodu LSH pro vyhledání podobných obrázků. Z výsledků testů vyplývá, že navržená metoda podává srovnatelné, a v některých případech dokonce lepší výsledky, než referenční metoda.

Protože navržená metoda používá globální popis obrazu, hodí se převážně pro vyhledání velice podobných obrázků. Oproti referenční metodě zavádí vyšší míru tolerance. Proto je metoda schopna vyhledat podobné obrázky i při relativně velké míře zkreslení oproti původnímu obrázku. Toto je demonstrované s využitím různě modifikovaných videí.

Na druhou stranu je navržená metoda pomalejší než metoda referenční, což vyplývá z vyšší náročnosti výpočtu navržených deskriptorů. Rychlost metody by bylo možné zvýšit efektivnější implementací. V této práci jsem se ale zaměřil zejména na modifikovatelnost (a přehlednost) a tomu jsem částečně podřídil i efektivitu výpočtu. Předpokládám, že výpočet deskriptoru by bylo možné změnou implementace zrychlit přibližně o 30 %.

Při dalším vývoji metody by bylo vhodné zefektivnit implemetaci výpočtu deskriptoru. Navržený deskriptor by bylo možné také použít v kombinaci s detektorem klíčových bodů i pro lokální popis obrazu. Deskriptor by tak nepopisoval celý obrázek, ale jen určité okolí klíčového bodu. Takovéto řešení by mohlo poskytovat zajímavé výsledky, jako například invarianci vůči rotaci obrazu, nebo změně pohledu kamery podobně, jako jiné lokální metody.

Literatura

[1] MIKOLAJCZYK, K.; et al. A Comparison of Affine Region Detectors. *International Journal of Computer Vision* [online]. 2005, roč. 65, 1-2, s. 43-72 [cit. 2011-12-20]. ISSN 0920-5691. DOI: 10.1007/s11263-005-3848-x. Dostupné z: <http://www.springerlink.com/index/10.1007/s11263-005-3848-x>

[2] KYSELÁK, M. *Multi-Index Approach for Similarity Searching* [online]. 2012 [cit. 2011-12-21]. Rigorózní práce. Masarykova univerzita, Fakulta informatiky. Vedoucí práce Pavel Zezula. Dostupné z: <http://is.muni.cz/th/98741/fi_r/>

[3] JOSEPHSON, W.; et al. Multi-Probe LSH: Efficient Indexing for High-Dimensional Similarity Search. In: *Proceedings of the VLDB Endowment* [online]. New York, NY: Association for Computing Machinery, 2007 [cit. 2011-12-21]. ISSN 2150-8097. Dostupné z: <http://www.cs.princeton.edu/cass/papers/mplsh_vldb07.pdf>

[4] SLANEY, M.; CASEY, M. Locality-Sensitive Hashing for Finding Nearest Neighbors [Lecture Notes]. *IEEE Signal Processing Magazine* [online]. 2008, roč. 25, č. 2, s. 128-131 [cit. 2012-01-06]. ISSN 1053-5888. DOI: 10.1109/MSP.2007.914237. Dostupné z: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4472264>

[5] ANDONI, A.; INDYK, P. E 2 LSH 0.1 User Manual. *Interface* [online]. 2005, 0.1, s. 0--22 [cit. 2012-01-06]. Dostupné z: <http://en.scientificcommons.org/54928730>

[6] CHUM, O; et al. Scalable near identical image and shot detection. *Proceedings of the 6th ACM International Conference on Image and Video Retrieval July 9-11, 2007, Amsterdam, the Netherlands* [online]. New York, N.Y: ACM Press, 2007, CIVR '07, s. 549--556 [cit. 2012-01-07]. ISSN 978-1-59593-733-9. Dostupné z: <http://research.microsoft.com/pubs/64603/civr2007.pdf>

[7] BOUMAN, C. A. Digital Image Processing [online]. 9. ledna 2012 [cit. 2012-01-09]. Dostupné z: <https://engineering.purdue.edu/~bouman/ece637/notes/pdf/Opponent.pdf>

[8] VAN DE SANDE, K. E. A.; GEVERS, T.; SNOEK, C. G. M. Evaluating Color Descriptors for Object and Scene Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* [online]. 2010, roč. 32, č. 9, s. 1582-1596 [cit. 2012-01-09]. ISSN 0162-8828. DOI: 10.1109/TPAMI.2009.154. Dostupné z: <http://www.science.uva.nl/research/publications/2010/vandeSandeTPAMI2010/vandeSandeTPAMI2010-EvaluatingColorDescriptors.pdf>

[9] PIHAN, R. *Vše o světle*: *5. Barevné modely* [online]. 2007, 23.02.2007 [cit. 2012-01-10]. Dostupné z: <http://www.fotografovani.cz/art/fozak_df/rom_1_05_colormodels.html>

[10] OTTO, G. *Graphics Short Course I*: *Color Theory and 2D Image Representation* [online]. 2000, 29.8.2000 [cit. 2012-01-13]. Dostupné z: <http://viz.aset.psu.edu/gho/sem_notes/color_2d/index.html>

[11] PASCALE, D. Review of RGB Color Spaces ...from xyY to R'G'B'. *The BabelColor Company* [online]. 2003, 2003-10-06 [cit. 2012-01-13]. Dostupné z: <http://www.babelcolor.com/download/A%20review%20of%20RGB%20color%20spaces.pdf>

[12] BRATKOVA, M; BOULOS, S.; SHIRLEY, P. ORGB: A Practical Opponent Color Space for Computer Graphics. *Kyungpook National University* [online]. 2009, 2009-03-14 [cit. 2012-01-13]. Dostupné z: <http://cilab.knu.ac.kr/seminar/Seminar/2009/20090314%20oRGB%20A%20Practical%20Opponent%20Color%20Space%20for%20Computer%20Graphics.pdf>

[13] MPEG-7 Overview. Chiariglione.org [online]. 2004 [cit. 2012-01-15]. Dostupné z URL: <http://mpeg.chiariglione.org/standards/mpeg-7/mpeg-7.htm>

[14] LOWE, D. G. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision* [online]. 2004, s. 91-110, 2004-01-05 [cit. 2012-01-16]. Dostupné z: <http://www.cs.ubc.ca/~lowe/papers/ijcv04.pdf>

[15] JACOBS, D. Image Gradients. *Class Notes for CMSC* [online]. 2005 [cit. 2012-01-16]. Dostupné z: <http://www.cs.umd.edu/~djacobs/CMSC426/ImageGradients.pdf>

[16] DALAL, N.; TRIGGS, B. Histograms of Oriented Gradients for Human Detection. *International Conference on Computer Vision \& Pattern Recognition* [online]. 2005, č. 2, s. 886-893, Leden 2005 [cit. 2012-01-16]. Dostupné z: <http://lear.inrialpes.fr/people/triggs/pubs/Dalal-cvpr05.pdf>

[17] DATAR, M.; et al. Locality-sensitive hashing scheme based on p-stable distributions. *Proceedings of the twentienth Annual Symposium on Computational geometry (SCG '04)* [online]. New York: ACM Press, 2004, s. 253--262 [cit. 2012-01-18]. ISSN 1-58113-885-7. DOI: 10.1145/997817.997857. Dostupné z: <http://www.cs.princeton.edu/courses/archive/spr05/cos598E/bib/p253-datar.pdf>

[18] JARKOVSKÝ, J.; LITTNEROVÁ, S. Vícerozměrné statistické metody. [online]. 2011 [cit. 2012-01-18]. Dostupné z: <http://www.iba.muni.cz/esf/res/file/bimat-prednasky/vicerozmerne-statisticke-metody/VSM-03.pdf>

[19] *Prostorové databáze*: *problémy indexace bodů v prostoru (algoritmy stromové, hašovací, kombinované)* [online]. 2009, 2009-08-17 [cit. 2012-01-18]. Dostupné z: <http://wp.soulwasted.net/msz/pdb/prostorove-databaze-problemy-indexace-bodu-v-prostoru-algoritmy-stromove-hasovaci-kombinovane>

[20] *Fotoslovník*. [online]. 2011 [cit. 2012-01-12]. Dostupné z: <http://phototv.cz/index.php?page=cataltxt&grouptxt=1&recid=53&lang=CZ>.

Seznam příloh

Příloha 1 Popis aplikace

Prostředí a knihovny

Deskriptory a aplikaci jsem implementoval v jazyce C++ s využitím knihoven OpenCV (http://opencv.willowgarage.com) ve verzi 2.3.1. Pro vývoj a překlad aplikace jsem použil nástroj Microsoft Visual C++ 2010 Express.

Funkce knihoven OpenCV jsem použil ke všem operacím s obrázky (čtení, vytváření, transformace) včetně zobrazování jednoduchého GUI výsledné aplikace. Dále jsem OpenCV využil například pro generování Gaussovy funkce a také jsem ve výsledné aplikaci využil implementace LSH které OpenCV taktéž obsahuje.

Pro procházení souborového systému (nalezení všech obrázků ve složce a podsložkách) jsem použil knihovnu „dirent.h“ (http://pubs.opengroup.org/onlinepubs/009695399/basedefs/dirent.h.html).

Implementace deskriptorů

Každý deskriptor je implementován jako samostatná třída.

- Referenční deskriptor – RefDescr
- Barevný deskriptor – ColHistDescr
- Gradient deskriptor – GradHistDescr
- Gradient+RG deskriptor – GradRGHistDescr

Parametry deskriptorů (dělení obrazu, překryv dlaždic a velikosti histogramů) jsou definovány v jejich konstruktoru.

Jedinou veřejnou metodou deskriptorů je metoda *GetDescr*, která vypočítá deskriptor předaného obrázku typu `IplImage` a tento deskriptor uloží do matice typu `Mat`.

```
void GetDescr(IplImage &img, Mat &descr);
```

Výsledná matice má jeden řádek o velikost rovné velikosti deskriptoru a je typu `CV_8U`, což je OpenCV ekvivalent k typu `unsigned char`.

Barevný deskriptor má navíc jeden nepovinný parametr konstruktoru a tím je volba barevného modelu (IO_1O_2, rgb, RGB), přičemž defaultně je použit model IO_1O_2.

Implementace aplikace

Aplikace byla vytvořena pro účely testování a demonstrace navržené metody. Je tedy implementována jako konzolová aplikace, která využívá prostředků OpenCV k zobrazování obrázků.

Poskytuje dvě základní funkce pro vytvoření indexu pro skupinu obrázků a k vyhledání podobných obrázků v tomto indexu na základě jiného obrázku.

Dále obsahuje tři pokročilé funkce, které slouží k porovnávání dvou videí a k detekci stejných, či podobných segmentů v těchto videích.

Za doplňkové funkce považuji funkci pro extrakci snímků z videa a funkci detekce segmentů v matici podobností spočítané pomocí jedné z předchozích funkcí.

Pro většinu vstupně-výstupních operací slouží v aplikaci třída `indexIO`, která obsahuje metody pro uložení a načtení parametrů indexu (`saveParams`, `loadParams`), metody pro uložení a načtení seznamu obrázků v indexu (`saveFilenames`, `loadFilenames`) a metodu, která načte všechny obrázky v určené složce a vypočítá jejich deskriptory:

```
void hashDir(vector<Mat> &descriptors, vector<string> &filenames, char* path)
```

Konstruktor třídy `indexIO` vyžaduje dva parametry typu string, které specifikují názvy souborů pro uložení parametrů a seznamu obrázků.

Pro vytvoření LSH indexu je použita třída z OpenCV – `flann::Index` – s potřebnými parametry – `flann::LshIndexParams`. Tato třída obsahuje vlastní metody pro uložení a načtení indexu.

Druhou důležitou částí aplikace je třída `VideoComparer`, která obsahuje metody pro načítání a porovnávání videí a metody použité pro detekci segmentů ve vypočítaných maticích podobností.

Tři metody třídy `VideoComparer` pro porovnávání videí:

```
void compareLSH(char *video1, char *video2, Mat & result, int countToFind)
```

- pro každý snímek z videa `video1` vyhledá tolik nejpodobnějších snímků ve videu `video2` kolik určuje hodnota `countToFind`. Vyhledané snímky zaznamená do matice podobností `result`.

```
void compareALL(char *videoOrig, char *videoModif, Mat &result)
```

- na rozdíl od `compareLSH` porovná každý snímek videa `video1` s každým snímkem videa `video2`. Výsledkem je matice podobností se záznamy typu `CV_8U`.

`int compareALLfloat(char *videoOrig, char *videoModif, Mat &result)`

- pracuje stejně, jako `compareALL`, ale výsledná matice podobností má záznamy typu `CV_32F`. Navíc vypočítá nejvhodnější hodnotu prahu, který umožní vhodné zobrazení výsledné matice a korektní detekci segmentů porovnaných videí.

Dále třída `VideoComparer` obsahuje metodu `hashVideo`, které pracuje na podobném principu jako metoda `indexIO::hashDir`, ale počítá deskriptory pro jednotlivé snímky videa.

Poslední metodou třídy `VideoComparer` je `findLines`, která pro matici podobností předanou jako typ `Iplmage` nalezne souvislé segmenty a vypíše jejich souřadnice (pozice ve videích). Matice podobností je nejdříve předzpracována (prahování -> dilatace -> median -> eroze) a poté je provedena Houghova detekce linií (implementace v OpenCV). Vyhledaná linie jsou následně protříděny. Jsou odstraněny čistě svislé a vodorovné linie (shodné segmenty ve videích se zobrazí jako diagonální linie). Velmi podobné linie a linie, které na sebe navazují jsou sloučeny.

Pro nalezené linie je vypočítána hodnota „score“ která udává, s jakou přesností byla linie nalezena.

Popis aplikace

Tato sekce se zabývá jednotlivými funkcemi aplikace, jejich parametry a ovládáním.

Vytvoření indexu

```
DP.exe --hash DIR [-V] [-N=xx] [-L=xx] [-DR|-DG|-DGC]
```

Vypočítá deskriptory všech obrázků ve složce `DIR` a jejích podsložkách a vytvoří LSH index, který uloží do souboru „database“. Parametry indexu jsou uloženy do souboru „database_params“ a seznam obrázků v indexu do souboru „database_filenames“.

Vyhledání podobných obrázků

```
DP.exe --find IMG [-C=xx]
```

Načte index uložený pomocí funkce „`--hash`“ a vyhledá a zobrazí 10 nejpodobnějších obrázků k obrázku `IMG`. Tento počet je možné změnit volitelným parametrem „`-C`“.

Porovnání dvou videí pomocí LSH

```
DP.exe --compareLSH VIDEO1 VIDEO2 OUTIMG [-N=xx] [-C=xx] [-L=xx] [-I] [-DR|-DG|-DGC]
```

Porovná dvě videa tak, že pro každý snímek z videa `VIDEO1` vyhledá určitý počet (defaultně 10) nejpodobnějších snímků ve videu `VIDEO2` výsledek zaznamená do matice podobností, kterou uloží jako obrázek `OUTIMG` (musí být zadán i s příponou – .jpg, .png, apod.).

Podle matice podobností navíc vyhledá segmenty videí, které jsou v obou stejné, či podobné a jejich pozice vypíše na stadardní výstup.

Porovnání dvou videí metodou hrubé síly

```
DP.exe --compareALL VIDEO1 VIDEO2 OUTIMG [-N=xx] [-L=xx] [-T=xx] [-I] [-DR|-DG|-DGC]
```

Porovná každý snímek z videa `VIDEO1` s každým snímkem videa `VIDEO2`. Výsledkem každého porovnání je hodnota 0–255 udávající míru podobnosti dvou snímků (0 znamená že dva snímky jsou stejné). Výsledkem je šedotónová matice podobností.

Následuje detekce podobných segmentů a uložení matice jako `OUTIMG`, stejně jako u předchozí funkce.

Porovnání dvou videí metodou hrubé síly s prahováním

```
DP.exe --compareCUSTOM VIDEO1 VIDEO2 OUTIMG [-N=xx] [-L=xx] [-I] [-DR|-DG|-DGC]
```

Pracuje stejně jako předchozí funkce s tím rozdílem, že výsledná matice podobností má hodnoty typu `float` a ty jsou před zobrazením a detekcí podobných segmentů prahovány pomocí vypočítaného ideálního prahu. Do obrázku `OUTIMG` je také uložena již prahovaná matice podobností.

Extrakce snímků z videa

```
DP.exe --extract VIDEO DIR [-N=xx] [-L=xx]
```

Doplňková funkce. Extrahuje z videa `VIDEO` každý N-tý snímek (defaultně N=10) a ukládá je ve formátu JPG do složky `DIR`.

Detekce segmentů v uložená matici podobností

```
DP.exe --detect IMG
```

Doplňková funkce. Provede detekci segmentů pro matici podobností uloženou v souboru `IMG`, které byla získána jednou z funkcí pro porovnání videí.

Volitelné prarametry

`-V`	Při vytváření indexu funkcí `--hash`, je místo složky s obrázky použito **video**. Funkce vyhledání podobných obrázků poté vypisuje čísla snímků videa, které jsou podobné obrázku `IMG`.
`-N=xx`	Při extrakci snímků z videa je použit pouze každý **N-tý snímkek**. Číslo `N` je zadané hodnotou `xx` (defaultně `xx=10`).
`-L=xx`	Hodnota `xx` udává **limit** maximální počet snímků extrahovaných z videa (defaultně bez limitu).
`-C=xx`	Hodnota `xx` určuje **počet podobných obrázků** vyhledaných pomocí LSH (defaultně `xx=10`).
`-I`	Zapíná **interaktivní mód** aplikace při porovnávání dvou videí. Vypočítaná matice podobností je zobrazena a pomocí dvou posuvníků je možné pohybovat se po této matici. Zároveň je zobrazen aktuální snímek obou videí (podle pozice posuvníku). Pomocí kláves + a − je možné měnit hodnotu prahu (krok 10) a sledovat změny na matici podobností. Klávesami `UP` a `DOWN` je možné měnit práh s krokem 1. Klávesou `ENTER` je možné provést detekci segmentů na základě právě zobrazené matice podobností. Interaktivní mód je ukončen stiskem klávesy `Esc`. Matice podobností je uložena s aktuální hodnotou prahu.
`-T=xx`	Hodnota `xx` definuje počáteční **hodnotu prahu**. Uplatní se pouze v interaktivním módu (defaultně `xx=30`).

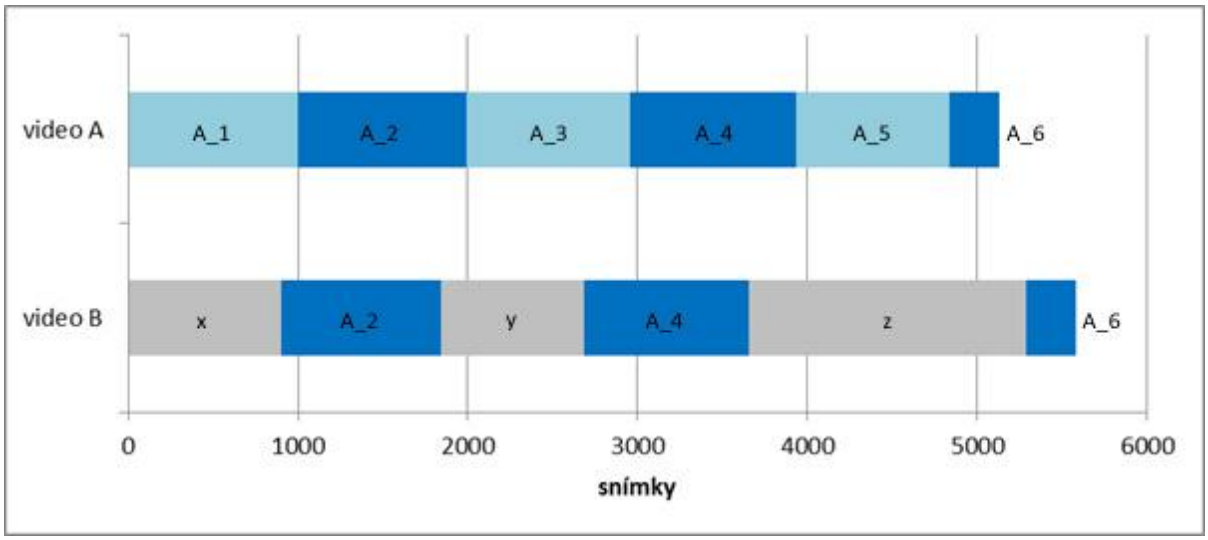
Typ deskriptoru. Implicitně je vždy použit Barevný deskriptor. To je možné změnit pomocí jednoho z následujících přepínačů:

`-DR`	Referenční deskriptor
`-DG`	Gradient deskriptor
`-DGC`	Gradient+RG deskriptor

Příklady použití

Vyhledání segmentů z videa A ve videu B

Video B vzniklo proložením segmentů z videa A a segmentů z jiného videa. Místo tří segmentů v originálním videu byly vloženy 3 jinak dlouhé segmenty z jiného videa. Složení videí ukazuje obrázek A.1.



Obrázek A.1 – Ilustrace složení videa B. Místo segmentů „A_1“, „A_3“ a „A_5“ z originálního videa A, byly ve videu B vloženy segmenty „x“, „y“ a „z“ z jiného videa.

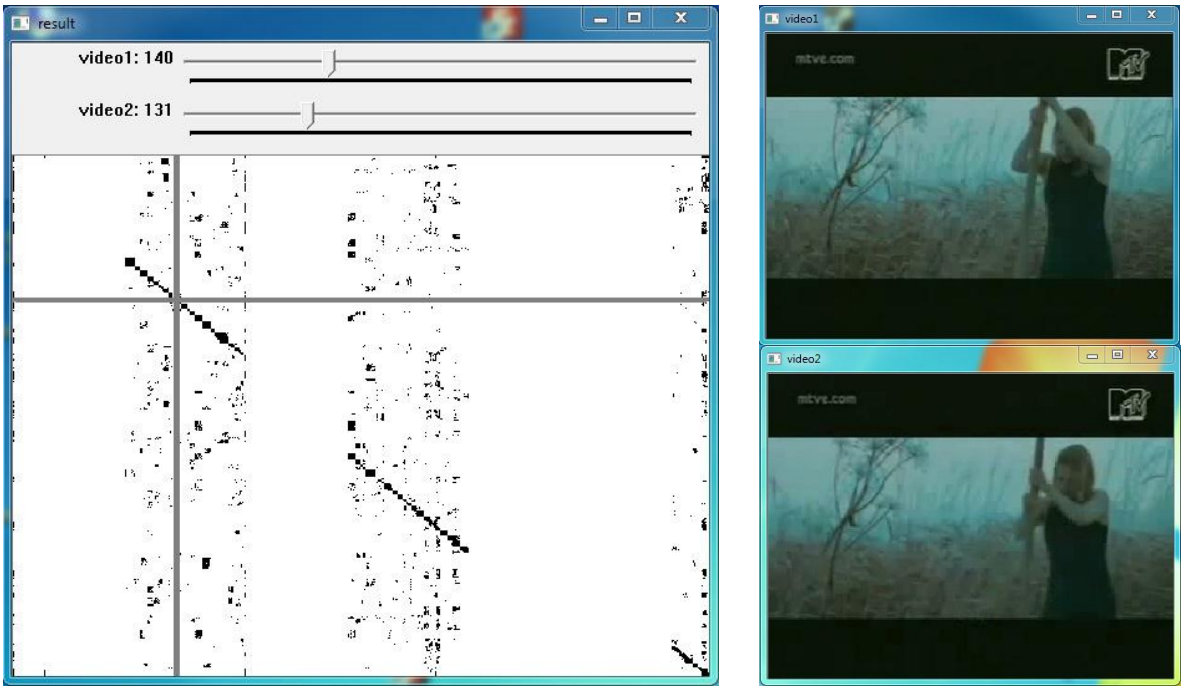
Aplikace byla použita pro porovnání videa A s videem B pomocí funkce „Porovnání dvou videí pomocí LSH“ s parametry N=10 (uvažuje se každý 10-tý snímek) a C=10 (pro každý snímek se vyhledá 10 podobných).

Obrázky A.2 a A.3 byly pořízeny v interaktivním módu aplikace. Na obou je vidět vypočítaná matice podobností. Svislá osa reprezentuje video A (označeno jako „video1“) a vodorovná osa video B (označeno jako „video2“).

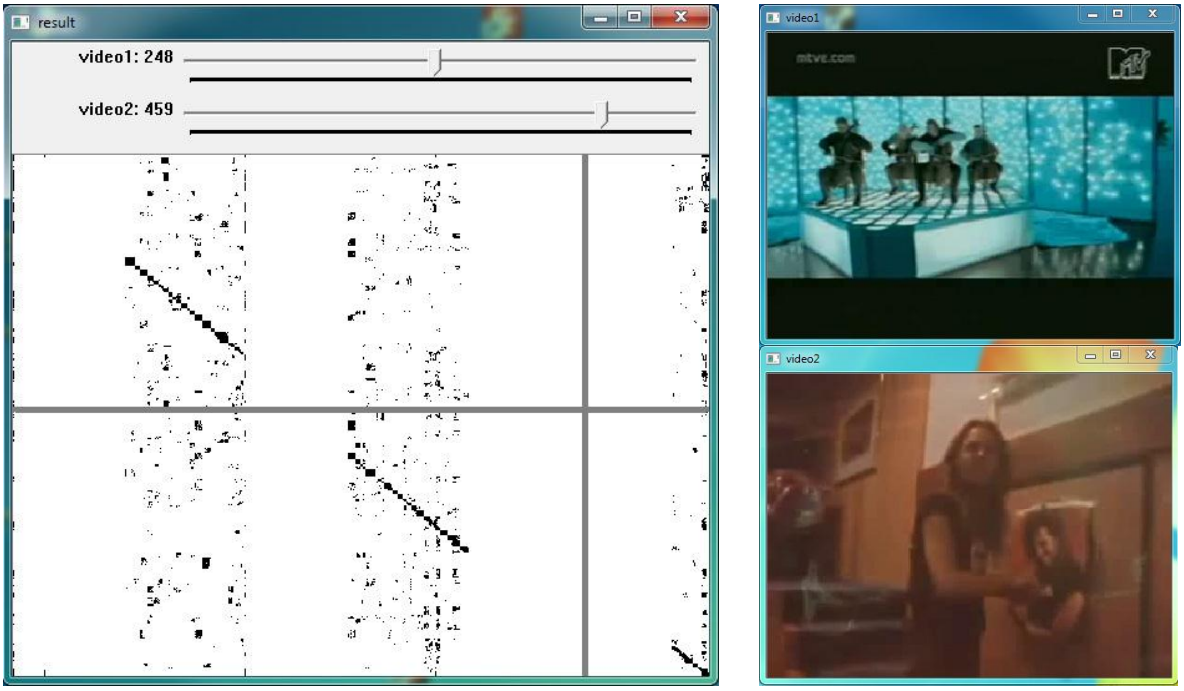
Na matici podobností jsou jasně patrné souvislé segmenty (černé diagonální linie). Tyto segmenty říkají, které segmenty z videa A odpovídají jakým segmentům ve videu B.

Obrázek A.2 ukazuje zobrazení snímků z každého videa, které leží na jedné z diagonálních linií. Je vidět, že nalezené snímky jsou opravdu velice podobné. Nejsou totožné, protože videa nejsou synchronizovaná (různé délky vložených segmentů) a při vzorku „každý 10-tý snímek“ nejsou při porovnávání videí použity stejné snímky. I přesto ale aplikace nalezla shodné segmenty.

Obrázek ilustruje zobrazení dvou snímků, které aplikace neoznačila za podobné. Je zřejmé, že snímky jsou opravdu různé a pocházejí z různých videí.



Obrázek A.2 – Ukázka dvou podobných snímků z porovnání dvou videí. Oba pocházejí ze stejného videa. Svislá osa reprezentuje video A a vodorovná osa video B.



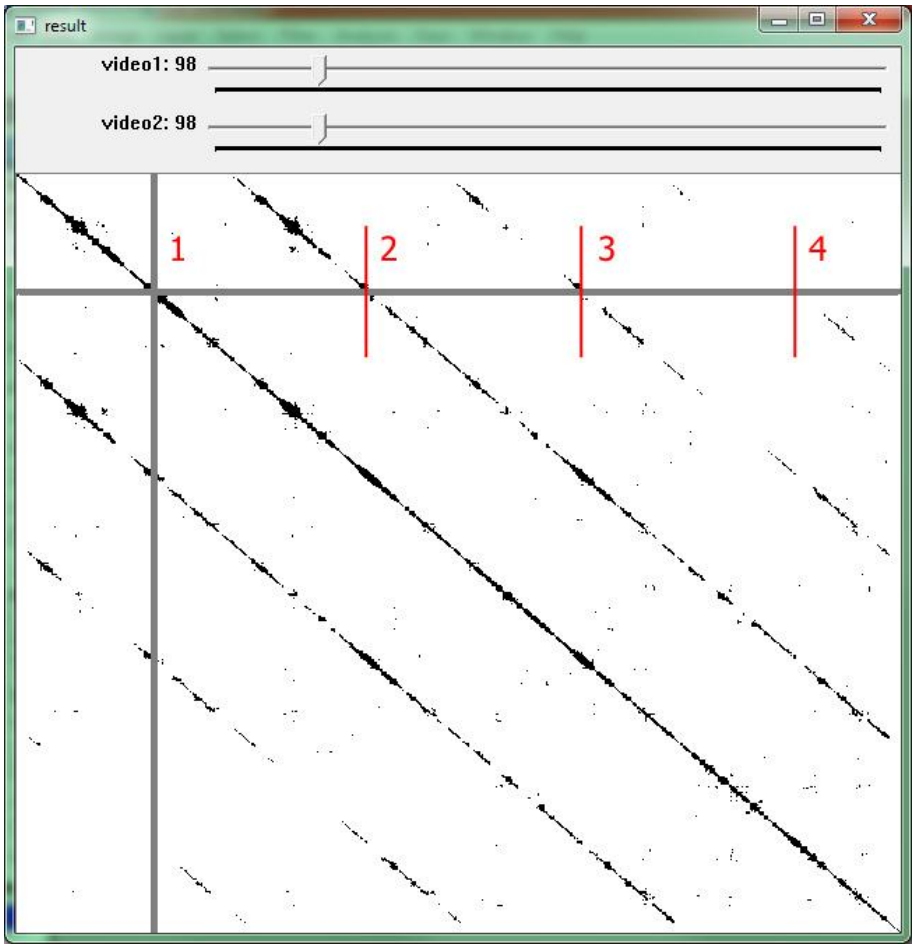
Obrázek A.3 – Ukázka dvou rozdílných snímků. Každý pochází z jiného videa. Svislá osa reprezentuje video A a vodorovná osa video B.

Nalezení opakovaných scén ve videu

Pokud necháme aplikaci provnat video samo se sebou (videa A a B jsou identická), můžeme nalézt vnitřní podobnosti tohoto videa. Taková vnitřní podobnost může být například nějaká scéna, která se ve videu opakuje. Například vícekrát opakovaný záběr s pouze malými změnami.

Pro demonstraci jsem vybral videoklip k písničce „Kylie Minogue – Come Into My World“. Toto video obsahuje jeden souvislý záběr kamery. Během videa zpěvačka 4krát obejde jakýsi okruh. Ve chvíli, kdy poprvé dokončí „okruh“, scéna se opakuje, ale zpěvačka je nyní na scéně dvakrát (první opakování). Stejným způsobem jsou duplikovány i další prvky ve scéně. Takto to pokračuje i při druhém a třetím opakování. Při posledním opakování je již scéna „přelidněná“ a tím se vytrácí podobnost oproti původní scéně.

Obrázek A.4 ukazuje vypočítanou matici vnitřních podobností. Hlavní diagonála ukazuje průběh videa od začátku až do konce. Daále jsou zde jasně patrná vedlejší diagonály, které ukazují nalezená podobné scény (v tomto případě opakování). První a druhé opakování je jasně patrné. Částečně patrné je i třetí opakování, ale jelikož je toto opakování již relativně dost rozdílné, tak není nalozeno s takovou jistotou.



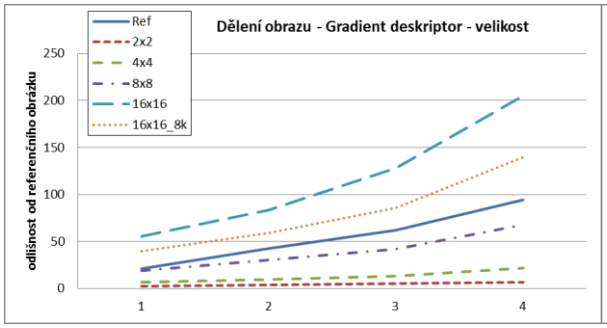
Obrázek A.4 – Matice podobností ukazující vnitřní podobnosti videa „Kylie Minogue – Come Into My World“. Diagonální linie ukazují výskyty stejných, či podobných scén.

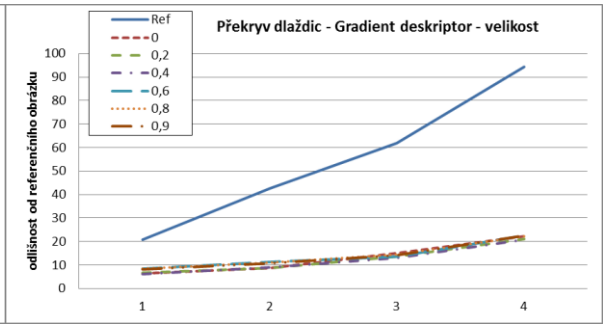
Na obrázku A.5 jsou zobrazeny snímky videa v místech označených na obrázku A.4. „Originální“ snímek z první scény (1) a snímky z jednotlivých opakování. Je vidět, že záběr kamery je stejný a zobrazuje stejné prostředí, ale při opakovaných scénách přibývají herci.

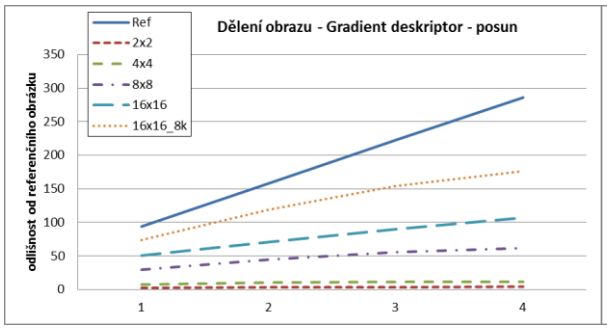
Obrázek A.5 – Odpovídající snímky z první scény (1) a z prvního, druhého a třetího opakování (2, 3, 4).

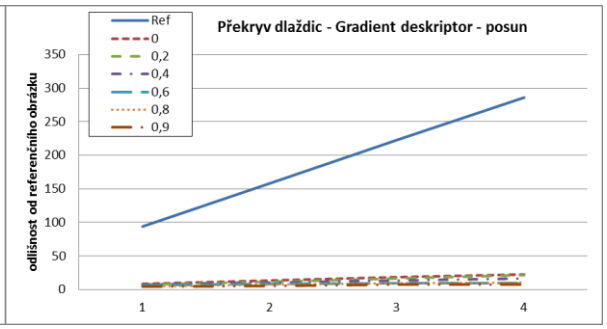
Příloha 2 Výsledky testů deskriptorů

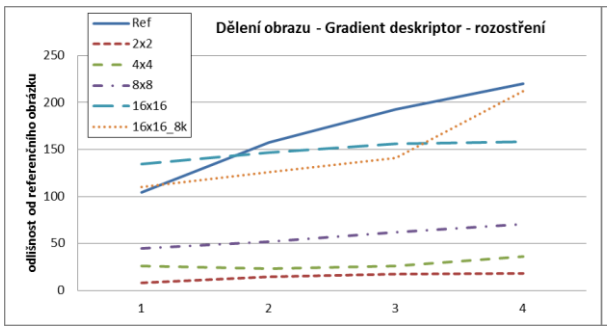
Gradient deskriptor

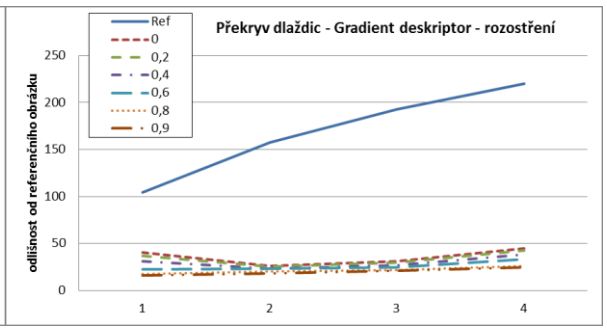


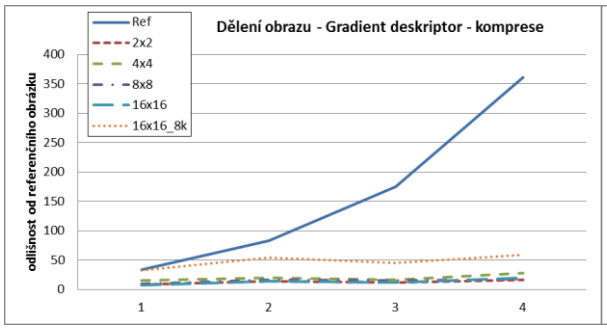


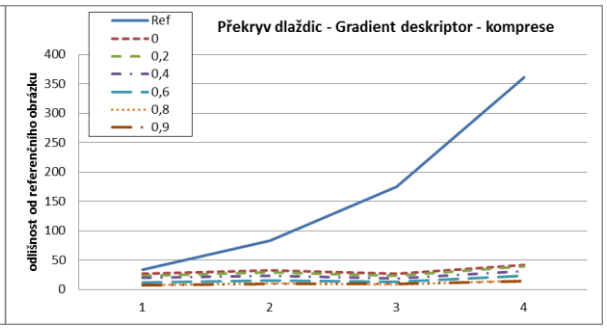


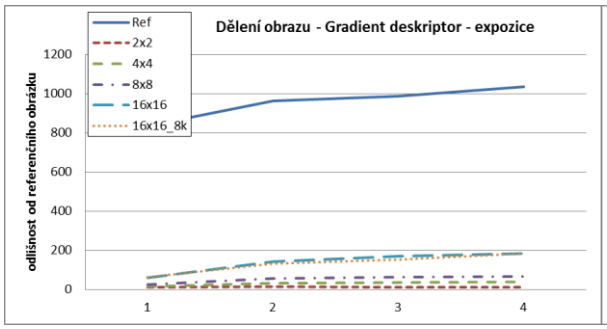


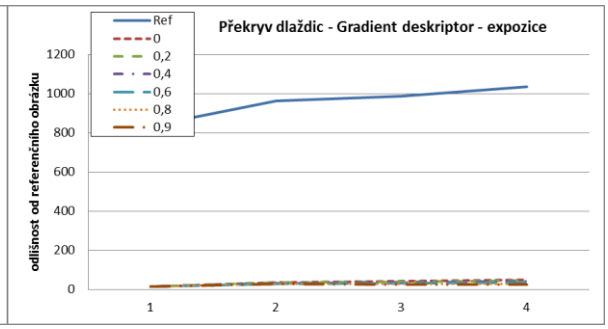


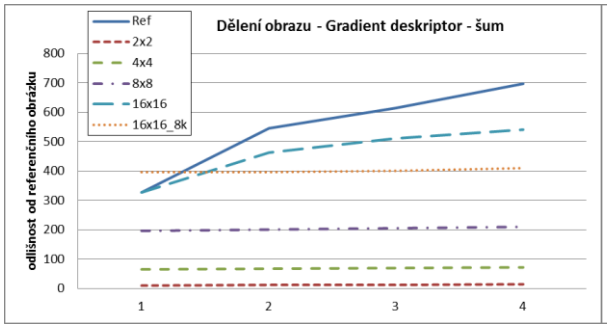


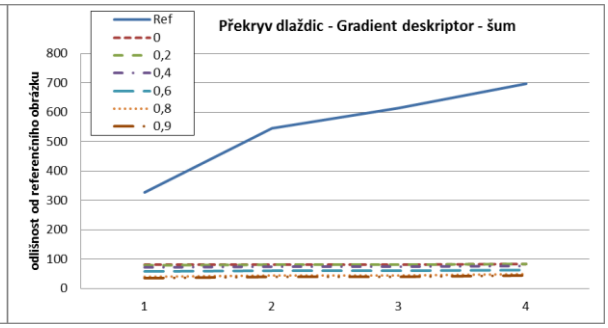


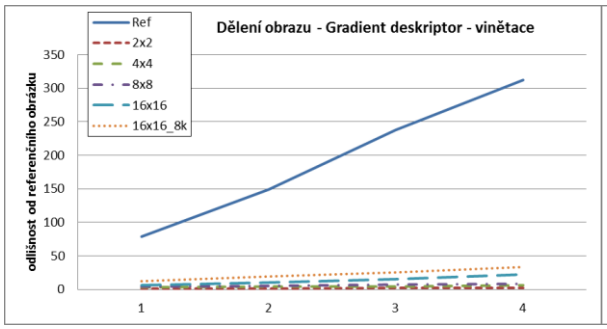


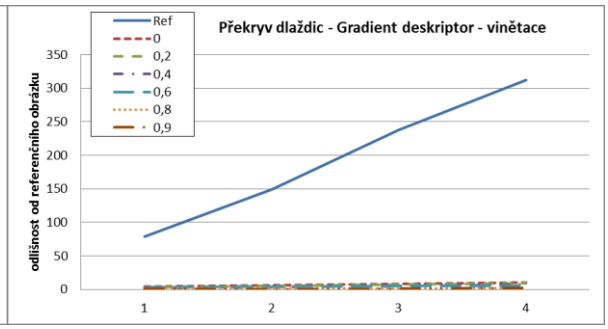




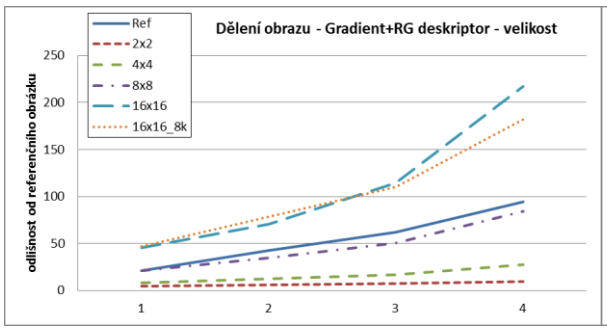


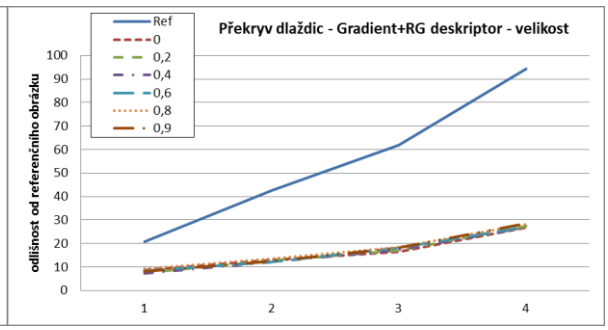


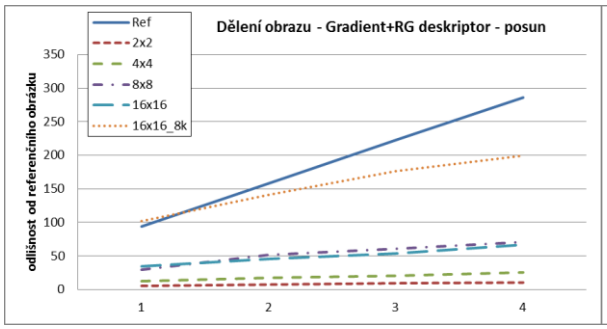


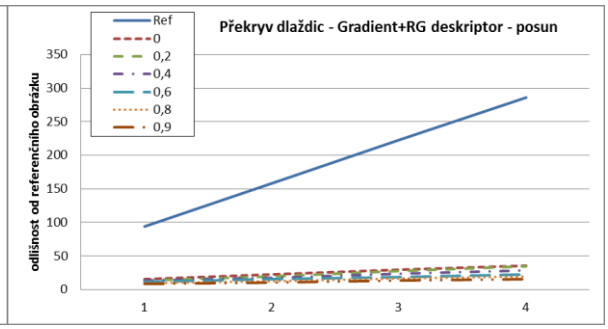


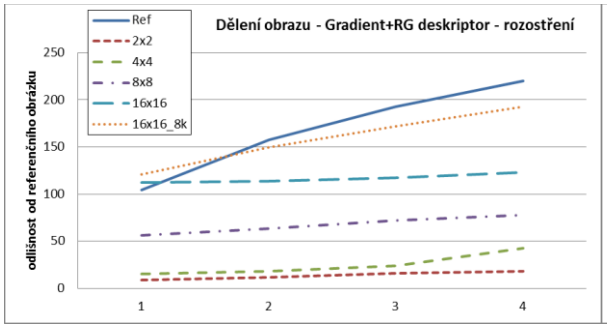
Gradient+RG deskriptor

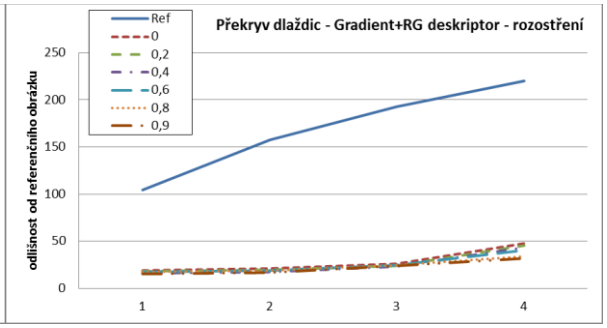


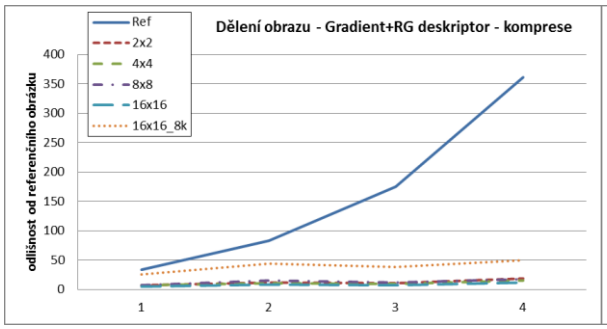


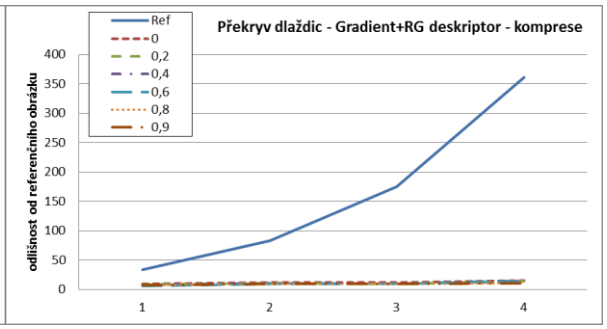


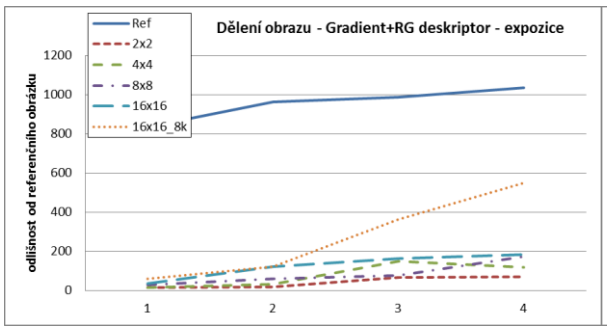


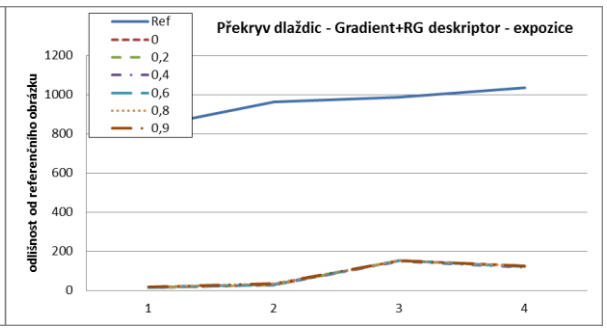

Dělení obrazu - Gradient+RG deskriptor - rozostření
Ref
2x2
4x4
8x8
16x16
16x16_8k
odlišnost od referenčního obrázku
0
50
100
150
200
250
1
2
3
4

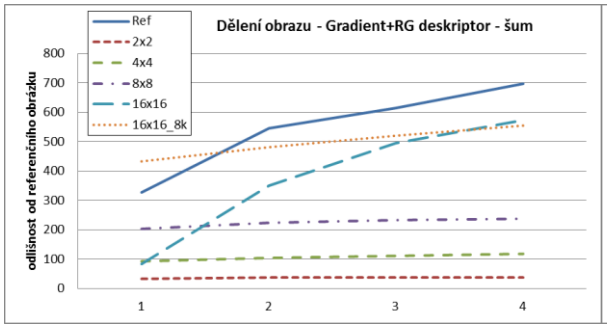

Překryv dlaždic - Gradient+RG deskriptor - rozostření
Ref
0
0,2
0,4
0,6
0,8
0,9
odlišnost od referenčního obrázku
0
50
100
150
200
250
1
2
3
4

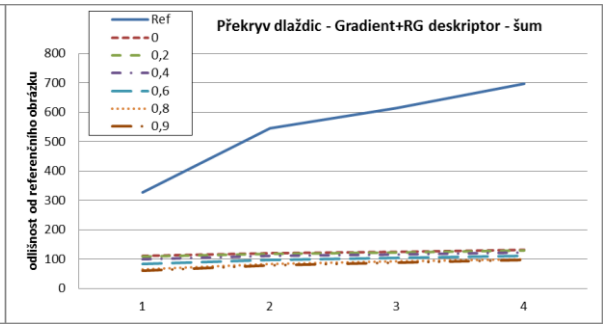

Dělení obrazu - Gradient+RG deskriptor - komprese
Ref
2x2
4x4
8x8
16x16
16x16_8k
odlišnost od referenčního obrázku
0
50
100
150
200
250
300
350
400
1
2
3
4

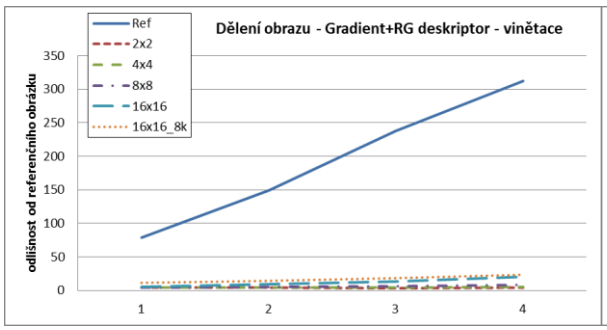

Překryv dlaždic - Gradient+RG deskriptor - komprese
Ref
0
0,2
0,4
0,6
0,8
0,9
odlišnost od referenčního obrázku
0
50
100
150
200
250
300
350
400
1
2
3
4

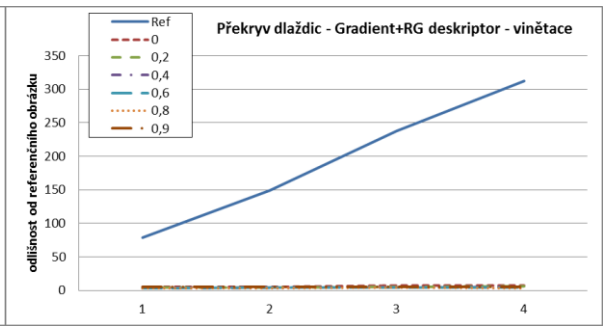

Dělení obrazu - Gradient+RG deskriptor - expozice
Ref
2x2
4x4
8x8
16x16
16x16_8k
odlišnost od referenčního obrázku
0
200
400
600
800
1000
1200
1
2
3
4


Překryv dlaždic - Gradient+RG deskriptor - expozice
Ref
0
0,2
0,4
0,6
0,8
0,9
odlišnost od referenčního obrázku
0
200
400
600
800
1000
1200
1
2
3
4


Dělení obrazu - Gradient+RG deskriptor - šum
Ref
2x2
4x4
8x8
16x16
16x16_8k
odlišnost od referenčního obrázku
0
100
200
300
400
500
600
700
800
1
2
3
4


Překryv dlaždic - Gradient+RG deskriptor - šum
Ref
0
0,2
0,4
0,6
0,8
0,9
odlišnost od referenčního obrázku
0
100
200
300
400
500
600
700
800
1
2
3
4


Dělení obrazu - Gradient+RG deskriptor - vinětace
Ref
2x2
4x4
8x8
16x16
16x16_8k
odlišnost od referenčního obrázku
0
50
100
150
200
250
300
350
1
2
3
4


Překryv dlaždic - Gradient+RG deskriptor - vinětace
Ref
0
0,2
0,4
0,6
0,8
0,9
odlišnost od referenčního obrázku
0
50
100
150
200
250
300
350
1
2
3
4

Příloha 3 Obsah DVD

DVD 1
|
|_ Aplikace
| |_ Spustitelná verze
| | |_Skripty
| |
| |_ Zdrojové kódy
|
|_ Data
| |_ Demonstrační obrázky
| |_ Demonstrační videa
| |_ Výstupy testů
|
|_ Knihovny
| |_ OpenCV
| |_ Dirent
|
|_ Média
| |_ Prezentační video
| |_ Plakát
|
|_ Text práce
|_ Zdrojové soubory

DVD 2
|
|_ Datová sada 1 (40 MB)
| |_ Expozice
| |_ JPEG komprese
| |_ Posun
| |_ Rozostření
| |_ Šum
| |_ Velikost
| |_ Vinětace
|
|_ Datová sada 2 (2,9 GB)
| |_ Originál
| |_ Komprese
| |_ Kvalita
| |_ Velikost+řez
|
|_ Datová sada 3 (1,1 GB)
|_ Originál
|_ Konkatenovaná videa

Složka „DVD1\Aplikace\Spustitelná verze\Skripty\“ obsahuje několik ukázkových skriptů (.BAT) pro jednodušší spuštění aplikace. Plnou funkcionalitu ale poskytuje spouštění aplikace z příkazového řádku, jak je popsáno v popisu aplikace.

Pro vyzkoušení aplikace je možné použít demonstrační videa a obrázky umístěné ve složce „DVD1\Data\“.

DVD 2 obsahuje datové sady použité při vývoji a testování metody.